energies

MDPI

Article

A New Hybrid Online and Offline Multi-Factor Cross-Domain Authentication Method for IoT Applications in the Automotive Industry

Haqi Khalid [1,*], Shaiful Jahari Hashim [1,*], Sharifah Mumtazah Syed Ahmad [1], Fazirulhisyam Hashim [1] and Muhammad Akmal Chaudhary [2]

[1] Department of Computer and Communication Systems Engineering, Faculty of Engineering, Universiti Putra Malaysia, Serdang 43400, Malaysia; s_mumtazah@upm.edu.my (S.M.S.A.); fazirul@upm.edu.my (F.H.)
[2] Department of Electrical and Computer Engineering, College of Engineering and Information Technology, Ajman University, Ajman 346, United Arab Emirates; m.akmal@ajman.ac.ae
* Correspondence: haqikhalid1@gmail.com (H.K); sjh@upm.edu.my (S.J.H.)

Abstract: Connected vehicles have emerged as the latest revolution in the automotive industry, utilizing the advent of the Internet of Things (IoT). However, most IoT-connected cars mechanisms currently depend on available network services and need continuous network connections to allow users to connect to their vehicles. Nevertheless, the connectivity availability shortcoming in remote or rural areas with no network coverage makes vehicle sharing or any IoT-connected device problematic and undesirable. Furthermore, IoT-connected cars are vulnerable to various passive and active attacks (e.g., replay attacks, MiTM attacks, impersonation attacks, and offline guessing attacks). Adversaries could all use these attacks to disrupt networks posing a threat to the entire automotive industry. Therefore, to overcome this issue, we propose a hybrid online and offline multi-factor authentication cross-domain authentication method for a connected car-sharing environment based on the user's smartphone. The proposed scheme lets users book a vehicle using the online booking phase based on the secured and trusted Kerberos workflow. Furthermore, an offline authentication phase uses the OTP algorithm to authenticate registered users even if the connectivity services are unavailable. The proposed scheme uses the AES-ECC algorithm to provide secure communication and efficient key management. The formal SOV logic verification was used to demonstrate the security of the proposed scheme. Furthermore, the AVISPA tool has been used to check that the proposed scheme is secured against passive and active attacks. Compared to the previous works, the scheme requires less computation due to the lightweight cryptographic algorithms utilized. Finally, the results showed that the proposed system provides seamless, secure, and efficient authentication operation for the automotive industry, specifically car-sharing systems, making the proposed system suitable for applications in limited and intermittent network connections.

Keywords: IoT applications; automotive industry; offline authentication; IoT-connected vehicles; cross-domain authentication

Citation: Khalid, H.; Hashim, S.J.; Ahmad, S.M.S.; Hashim, F.; Chaudhary, M.A. A New Hybrid Online and Offline Multi-Factor Cross-Domain Authentication Method for IoT Applications in the Automotive Industry. *Energies* **2021**, *14*, 7437. https://doi.org/10.3390/en14217437

Academic Editor: Manuel Moreno

Received: 18 July 2021
Accepted: 23 September 2021
Published: 8 November 2021

1. Introduction

The Internet of Things (IoT) paradigm has profoundly affected the automotive industry and its long-term prospects [1,2]. Traditional vehicle-to-everything (V2X) technologies are evolving into the Internet of Vehicles (IoV) to support emerging advanced vehicular applications, including intelligent transportation systems (ITS) and autonomous vehicles [3,4]. The global market for autonomous driving is expected to reach $173.15 billion by 2030, as reported by [5]. Numerous automobile businesses, such as ArgoAI, Audi, Baidu, Cruise, Mercedes-Benz, Tesla, Uber, and Waymo, have made significant investments in this domain. Customers, suppliers, and service providers will benefit from unprecedented data collecting, easy connectivity, location-based utilities, personalized insurance benefits, intelligent diagnostics, and assisted driving as

the Internet of Things (IoT) is introduced to cars [6,7]. Although these opportunities are essential somehow, they are only as good as their weakest link [8]. According to Statista, more than 26 billion IoT-connected devices are in use today, with that number expected to rise to more than 75 billion by 2025. As the connected car's knowledge, facilities, and environment evolve, so do the risks and exposures that come with it [9,10]. According to Intel predictions, autonomous vehicles may generate and consume 5 TB of data each hour of driving, with automotive cameras and radars alone generating data at rates of 20–40 Mbps and 100 kbps, respectively, [4]. Additionally, data are derived from crowdsourced traffic networks such as Google Waze. Processing such large amounts of data generated by hundreds of thousands of vehicles requires enormous computational capabilities and effective machine learning techniques to extract crucial information that must be produced for each driver [5]. As electrical vehicles (EV) are being pushed instead of internal combustion engine (ICE) technology, it is only natural that the connectivity technology is becoming more sophisticated. Furthermore, vehicles have become increasingly autonomous to provide more safety and convenience to the users. However, the increases in sophistication and autonomy need to be balanced with comprehensive security, regardless of the condition and location of the car. Central to this connected vehicle initiative is the owner's smartphone. It is used to control, monitor, and secure the vehicles and other necessary interactions with their environment, for example, automatic parking, toll services, gas and charging stations, automakers' service networks, and other driving facilities, as shown in Figure 1.

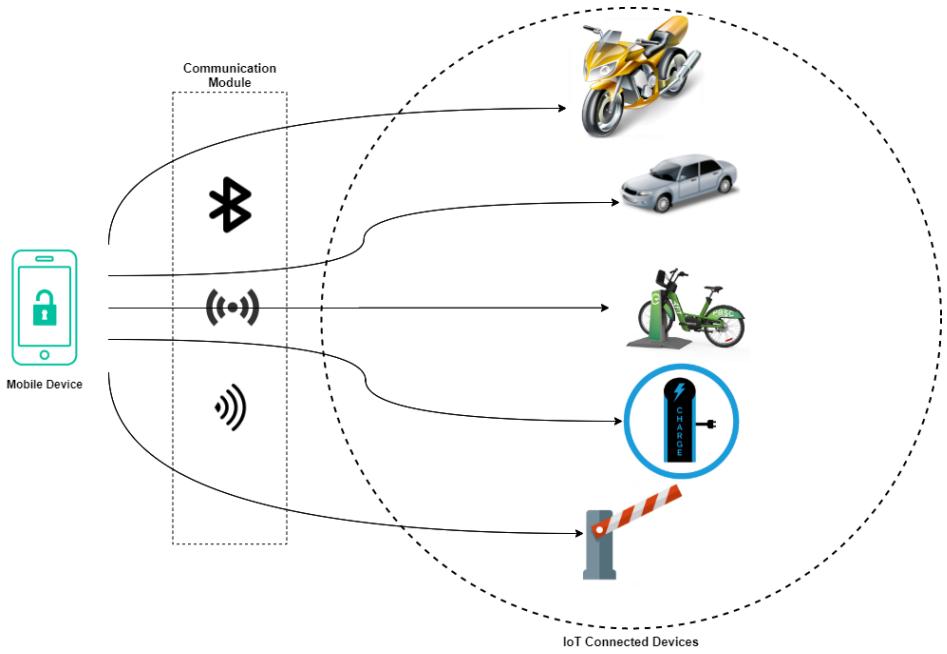

Figure 1. Smartphone-centric vehicle control, monitoring, and security facilities.

Furthermore, automakers emphasize their connected features, which range from on-board Wi-Fi to mobile applications that monitor locks and even start vehicles. The novelty of these "smart" features frequently outweighs the negative consequences in these situations. Thus, what happens if a customer's phone is lost or stolen? Is there adequate protection and authentication in place to prevent their car from being stolen as well? What happens if the customer's Internet connection is lost? Both are things to consider. Over the last few decades, vehicle networks have progressed from essential communications inside a vehicle using the CAN Bus technology (control area network) [11]. However, the issue with CAN Bus is that it was not built with safe communications in mind, and attempts to introduce secure certificate-based authentication to CAN Bus devices have largely failed. A cyberattack is a pernicious endeavor to breach the data or systems of an individual or a specific organization [12]. Several cyberattacks are possible now, such as

information fraud, blackmailing, malware, man in the middle, server hijacking, spamming, trojans, phishing, denial-of-service, and so on [13]. Replay attacks are one of the many steppingstones for car hacking [14]. Replay attacks happen when malicious users record the signals between two parties. Replay attacks can occur when messages between vehicles are not encrypted and authenticated. The receiver will verify the sender as a legitimate user; while this exchange is happening is when the malicious user comes into play [15]. This, in return, makes the receiver think that this is the original sender. However, it is the malicious user gaining access to unauthorized information using a "replay" of the authenticated user's signal [15,16]. Bluetooth and Near-Field Connectivity (NFC) are the common ways for a vehicle and a smartphone (NFC) to interact [17]. Bluetooth Low Energy (BLE) is part of the Bluetooth 5 specifications, designed to use low electricity. BLE also has the advantage of not needing system pairing. BLE is exceptionally convenient because of this. NFC is also convenient, but proximity is necessary for functioning. It is less convenient than BLE but probably more secure due to the proximity limitation, unlike BLE, which can be potentially exposed replay attack if adversaries are nearby. As a result, BLE is a common choice for real-world use. The low transmission speeds of BLE and NFC constitute a disadvantage, and theoretical bandwidth is often not achieved in practice [18]. Maintaining a continuous network connection, however, is often unlikely or unwanted due to higher costs. Moreover, in a world where most smartphones always have an Internet connection, a protocol where all steps can be completed without a network connection is necessary. However, offline authentication is used for in-person transactions where access is inaccessible or unnecessary. Thus, there must be a way of checking that a person is who they claim to be without reference to other systems (remote identity databases, online services, etc.), and if possible, that the credentials they present are genuine. Accessing the vehicle in such a network connectivity shortage makes the development of IoT-connected vehicles undesirable [19]. On the other hand, IoT services are used widely nowadays in car maintenance applications, door unlocking, and even car-sharing services. Hence, the network connection's availability should not be an issue when offline authentication exists. Several works [16,20–23] tried to cope with a specific type of attack, which can be a limitation. In contrast, several others were designed to handle a specific challenge, such as authentication [24], privacy [25], or localization [26]. Therefore, unlike other solutions, this study proposes a hybrid online–offline authentication scheme for Industrial IoT-connected vehicles in the real world, where connectivity can be unreliable or intermittent due to many reasons and circumstances. The scheme utilized an AES-ECC algorithm for secure communication and efficient key generation management. The Kerberos workflow is used to enable users to book the car in online mode. The one-time password (OTP) is also added to the offline authentication to allow the user to access offline mode when the connectivity is unavailable in regions with poor network availability.

2. Related Works

Security is a big problem in IoT-connected sharing systems. Many public-key cryptosystems have been proposed for low-function devices. Addobea et al. [27] proposed the MHCOOS, a bilinear pairing-based offline–online signature scheme for M-Health applications. However, bilinear pairing necessitates high pairing and map-to-point function operations, which are inappropriate for resource-constrained IoHT devices. Under the RSA assumption, Yu and Tate [28] proposed an efficient online–offline signature scheme proven to be secure without a random oracle. At the trapdoor, they did not use the hash function. As a result, their scheme did not have to deal with the second key pair, and they did not have to include the random commitment attribute in their signature. However, since the RSA cryptosystem is based on hard problems and has a high computational cost, the proposed scheme is not affordable for constrained IoT devices. Using bilinear pairing, Wu et al. [29] proposed a competitive online–offline signature scheme. The theoretical Diffie–Hellman assumption in the random oracle model is linked to the model's security. Shamir and Tauman [30] used chameleon hash functions based on an ordinary digital

signature to create an efficient online–offline signature scheme in 2001. According to the original scheme, the major scale and signature sizes were reduced in the proposed scheme. To improve device security, they added a new hash function called the trapdoor hash function to their model. The receiver obtains a hash collision and extracts trapdoor information from the signer, which is the signer's secret key [31]. The signer uses the same hash value to get two signatures on two different messages. The suggested scheme, on the other hand, employs many chameleon hashes values for different messages. This problem is the primary chameleon hashing disclosure issue. D. Liu et al. [32] devised an effective identity-based online/online signature scheme that avoids the main escrow issue by implementing the CLC concept. PKG generates only a partial private key, while the user generates the other partial private key, making the complete private key. As a result, the PKG is unaware of the user's private key in its entirety. The system, however, suffers from a high computational burden due to issues with identity-based signature computation. Dmitrienko et al. [33] proposed a free-float, offline-enabled car-sharing control scheme. This protocol is built on symmetric encryption, protected elements for storing private credentials, and a single car-sharing provider for access rights management. Dmitrienko et al. [34] suggested a generic access control scheme based on NFC-enabled devices that includes offline validation and authorization delegation. However, several proprietary protocols are used in this work. SePCAR is a smart car access control protocol. On the other hand, this protocol is more concerned with user privacy than with bandwidth efficiency [35].

S. Hass et al. [36] presented an offline authentication system for direct access to Industrial mobile robots (IMRs) in production facilities using one-time passwords (OTP), protected components, and smart cards. The authentication method offers two separate authentication modes to provide more flexibility. Authentication modes are used to ensure that passwords are generated from the identity of a particular IMR or group of IMRs that is accessed rather than a person's credentials. As a result, the system is susceptible to a replay attack, also known as a man-in-the-middle attack. Jia-Li Hou et al. [28] proposed a sensor-based offline–online authentication architecture for IoT healthcare systems. A robust co-existence proof protocol and a stable single sign-on authentication scheme were defined. The proposed approach combined the SSO technique with a one-way hash function and a random nonce to provide protection and performance. Following that, Li and Xiong [37] devised a safe scheme for achieving confidentiality, honesty, authentication, and nonrepudiation in a single logical stage. The proposed method divides encryption into two stages, one online and one offline, allowing a sensor node in an identity-based cryptosystem to communicate with an Internet host. Consequently, this scheme effectively incorporates WSN into IoT [38]. Saeed et al. introduced a CLC to PKI online–offline HS scheme for IoT in 2017 [39]. Besides, the authors put the scheme into practice in the fields of healthcare and smart grids. Via the signature of a certificate authority, the PKI connects each public key to its corresponding user identity (CA). PKI systems are not ideal for use in industrial IoT because managing certificates is a difficult job. Vonoth et al. 2020 [40] suggested a stable multi-factor authenticated key agreement scheme for IIoT to enable approved users to remotely access to sensing devices. They devised an authenticated key agreement scheme for simultaneously accessing multiple sensing devices and establishing a mutual session key between them. Unfortunately, due to secret-sharing technology, the proposed scheme has a high computing cost [41].

The use of blockchain-based offline authentication for a smart lock was proposed by Han et al. [42]. Centered on blockchain technology, this paper proposed a non-interactive end-to-end offline authentication scheme (BC-SNOA). Instead of using the pad in real-time, the BC-SNOA scheme uses it once. The subscriber only needs to pick the relevant information (reservation person's mobile phone number, conference name, meeting length, etc.) in this decision, and the Internet booking service system will calculate an encrypted token string and create a QR code from it. It employs the proof-of-work consensus algorithm, which entrusts the hard work to the miners. The miners are rewarded for solving difficult mathematical problems. Furthermore, the decentralized construction systems

suffers from single point failure. The high energy consumption renders these complex mathematical problems unsuitable for real-world applications [43,44]. Nevertheless, the advantages and disadvantages of the existing authentication schemes that work in online and offline modes are shown in Table 1.

Table 1. Comparison of the existing online–offline authentication schemes.

Ref.	Issue	Method	Advantages	Disadvantages
[10]	Offline identity guessing attack, location spoofing attack, and replay attack,	CDH	prevents offline guessing attacks	Intruder has the capability to disrupt integrity, authenticity, confidentiality.
[28]	Devices with limited computing capabilities.	RSA algorithm	Improve the efficiency of both online and offline phases.	Not feasible for IoT devices with limited resources.
[29]	Attacks based on existential forgery on adaptively chosen messages	Bilinear pairing	Secure from forgery attacks.	High computation due to Bilinear pairing.
[30]	The trade-off between the size of the keys and the complexity.	Trapdoor hash function	Secure against adaptive chosen message attacks.	For different messages, several chameleon hashes values are used.
[32]	Overcomes the key escrow problem,	Bilinear Pairing and MTP function.	Resolve the key escrow problem.	High computation due to Bilinear pairing.
[33]	Online communication shortcomings.	Identity-based encryption	Allows the car to expand its services to areas without reliable network coverage.	Suffers from various passive attacks.
[34]	Leakage and unintended manipulation of security-critical data	Identity-based encryption	Provides a two-line defence against software attacks	Requires high computation and communication costs.
[35]	Allows users to share their cars conveniently without sacrificing their security and privacy.	AES	Ensure that messages sent between vehicles and VPKI servers are unlinkable.	The security of the protocol was not proven.
[36]	Robots suffer from higher safety risks than	One- Time Passwords (OTP).	Secured method for offline authentication on mobile robots	Vulnerable to replay attack, man-in-the-middle attack.
[45]	Limited computational resources of low-cost IoT based devices make the design of security components for such devices difficult.	Single sign-on (SSO)	Use the unitary token to access different services.	Vulnerable to impersonation and modification attacks.
[37]	Insecure communication between a sensor node and an Internet host.	Identity-based cryptography	Works against adaptive chosen-ciphertext attacks.	High complexity due to the Bilinear pairing.
[39]	Design a heterogenous scheme.	Certificateless signcryption	Overcome the key escrow	Not suitable for use in industrial IoT.
[42]	Secure end-to end security.	Blockchain	Secure transmission	High complexity, energy consumption, single point failure.

As mentioned above, specific online–offline authentication schemes were proposed to obtain a valuable service to the customers. Despite this, most of these schemes have not been proven reliable. The few proven to be secure (under conventional cryptographic assumptions) are too slow for many practical applications. Furthermore, the vehicle could book or deal with IoT devices located in a different location out of their current zone. Although there have been many papers on identifying and mitigating each type of threat, the lack of design support still challenges security engineering for development. These schemes are based on open network architecture and necessitate ongoing online services. Electromagnetic attacks, vulnerability scanning infiltration, network eavesdropping, and service system and database attacks can be used by hackers to break into rooms, steal

money, steal data, and disrupt services, posing a danger to the entire industry. The proposed solutions fail to provide essential security features, such as vehicle anonymity and cross-domain authentication. They are also prone to various known attacks, including man-in-the-middle, replay, unlinkability, and vehicle impersonation and modification attacks. Moreover, these schemes incur heavy computation and communication costs, making them impractical to adapt to real-time scenarios. We propose a hybrid online–offline multi-factor cross-domain authentication scheme for the industrial IoT environment to address these issues. The proposed scheme involves online and offline phases, and the first phase is to enable the user to book the vehicle that belongs to another domain. Otherwise, the advantages of AES are faster execution in both the hardware and software; it meets the latest that is required by the United States' and international standards; it is also more secure to use; lastly, it supports larger key sizes than other algorithms [46]. The disadvantages of AES are that it is challenging to know the details of the process because the encryption is patented, and it will be difficult to decrypt the data (Cipher text) if the secret (private) keys are lost [47]. Therefore, our study added the ECC algorithms for efficient key management. The TOTP generator is bound to the user's device (for example, mobile device, or hardware token). Suppose this device is stolen, lost, or breaks. In that case, the association between the service provider and the TOTP generator is lost, and the service provider needs to re-issue a TOTP authenticator for the user. Thus, the study provides multi-factor authentication using user biometric to unlock the car via a biometric reader deployed in the car. The offline application is applied when the network connectivity is unavailable based on the OTP generated before the online booking phase.

3. Functionality and Security Goals

Based on the literature, an unquestionable requirement in terms of security and functionality has still not been met in previous studies or any IoT-connected devices. Therefore, to ensure the security and functionality of the proposed scheme, the following requirements are considered to provide an efficient and secure authentication scheme:

- Mutual Authentication: To ensure the effectiveness of all participants, the vehicles and the servers need to authenticate each other.
- Offline Authentication: The car or any IoT-connected devices solutions depend on connected vehicles, which restricts their functional areas to areas with a stable network link. To address this restriction, we require users to authenticate offline during car (un)locking, allowing car-sharing services to go to areas with less secure networks or no network access at all.
- Vehicle anonymity: When a user uploads his ID to the medical server, adversaries should not obtain the user's identity during the registration process.
- Low Computing Cost: The IoT devices are resource constrained devices with low power, so the authentication schemes must be designed with low computational costs to suit the IoT devices' requirements.
- Integrity: The vehicle may authenticate each message to ensure that it was not tampered with by an adversary.
- Confidentiality: Passive attacks such as eavesdropping or traffic monitoring should not be able to access vehicle data. As a result, only designated individuals have access to or use vehicle data.
- Forward Secrecy: The proposed protocol should provide backwards and forwards confidentiality to ensure the protection of messages exchanged in previous and future communications. Even if the adversary obtains the current session key, he cannot get the session keys created in prior and subsequent sessions.
- Resistance Against Attacks: In-vehicle and device communication, any newly developed authentication scheme must be resistant to masquerade attacks, alteration attacks, replay attacks, and man-in-the-middle attacks.
  - Replay attacks: In IoT-connected vehicles, replay attacks should be prevented by using timestamps and random numbers in the transmitted messages [48].

- Man-in-middle attack: An adversary intercepts the messages sent between the vehicle and server and replaces them with messages.
- Offline password guessing attack: In this attack, an attacker can employ some of the intercepted information, such as keys, or the self-generated parameters, to guess the user's password [49]. These attacks can never be "prevented," but protocols can be made secure against such attacks.
- Server spoofing attack: This attack can be solved entirely by providing mutual authentication between vehicle and server.
- Privileged insider attack: When the server needs to retain the user's password for later authentication, the keys are probably being stolen by the adversary because the server can find out the patient's new password.
- Denial of service (DoS) attack: Services are denied to the attackers by the automative users/vehicles and the servers [50].
- Impersonation attack: A dishonest user can easily impersonate another legal user.

4. Cryptography Materials

This section gives a brief overview of elliptic curve cryptography [49]. The ECC's security is based on solving the elliptic curve discrete logarithm problem (ECDLP). It can achieve the same degree of protection with a smaller key size [51]—the use of ECC with a more minor key size results in significant cost savings. Furthermore, ECC's smaller key sizes make it easier to design faster cryptographic operations that can run on small chips with limited memory. This is suitable for systems with limited resources, since it reduces power consumption and heat output. Furthermore, the combination of the AES-ECC method is briefly discussed and gives the steps of the method workflow. The AES is used to encrypt the message before it is transmitted using the receiver public key. It was chosen as an asymmetric key algorithm because of its highly secure symmetric algorithm, and it is easy to implement without affecting the complexity. As a result, AES-ECC is well suited for smart devices operating in resource-constrained situations [52].

4.1. Elliptic Curve Cryptography (ECC)

Let E/F_p be a set of elliptic curve points over a prime field F_p, defined by the following non-singular elliptic curve:

$$y^2 \; mod \; p = (x^3 + ax + b) \; mod \; p \tag{1}$$

where $x, y, a, b \in F_p$ and $(4a^3 + 27b^2) \; mod \; p \neq 0$. A point $P(x, y)$ is an elliptic curve point if it satisfies (1), and the point $Q(x, -y)$ is called the negative of P, i.e., $Q = -P$. Let $P(x_1, y_1)$ and $Q(x_2, y_2)(P \neq Q)$ be two points on (1); line l (tangent to the curve (1) if $P = Q$) joining the points P and Q intersects the curve (1) at $R(x_3, -y_3)$ and the reflection of it concerning x-axis is the point $R(x_3, y_3)$, i.e., $P + Q = R$. The points E/F_p together with a point O, called "point at infinity" or "zero points," make an additive elliptic curve cyclic group G_p; i.e., $G_p = (x, y) : x, y \in E/F_p$ and $(x, y) \in E/F_p \cup \{\}$ of prime order p. The scalar point multiplication on G_p is defined as: $k \cdot P = P + P + \cdots + P(k \; times)$. A generator point $P \in G_p$ has order n if n is the smallest positive integer and $n.P = O$.

4.2. AES-ECC Algorithm

This section discusses the AES-ECC workflow. The AES key is secured in this algorithm by encrypting it with the ECC key without increasing the complexity or cross-encrypting the AES and ECC keys [53]. The workflow diagram of the AES-ECC algorithm is shown in Figure 2, and the steps are illustrated as follows:

1. Input data, i.e., username, password, and a biometric of the user using a smartphone.
2. The data are hashed using the SHA-2 function to generate a hash value for data summary.
3. It generates a digital signature using a private sender key K_s and an ECC digital signature.

4. Using the private key of the AES, the sender encrypts the digital signature and the data that need to be sent result in data ciphertext and signature ciphertext.
5. The sender then encrypts the AES private key using the ECC encryption module to generate the key ciphertext. Then, it sends the ciphertext via the Internet.
6. Upon receiving, the receiver uses his private key to decrypt the AES key; then, it decrypts the data ciphertext and signature ciphertext using the AES key.
7. Based on the sender's public key, the receiver verifies the signature to summarize the received data and the hash value using the SHA-2 function. If the value is the same, then the data are valid; otherwise, the session is ended.

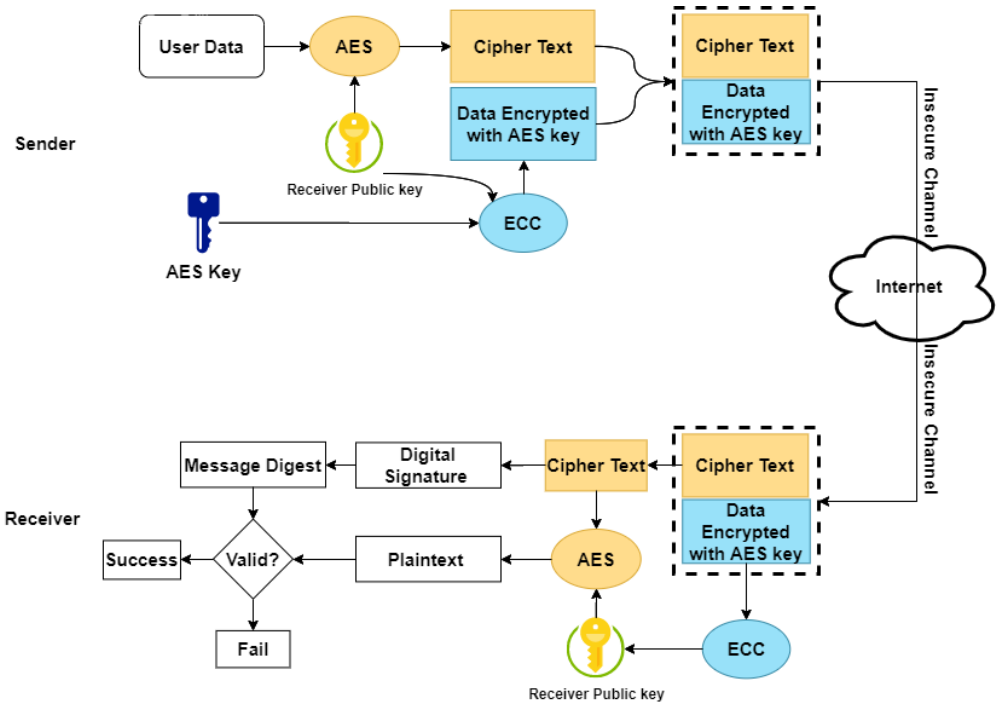

Figure 2. The workflow diagram of the AES-ECC algorithm.

5. Proposed Scheme

This section proposes a new and secure multi-factor cross-domain authentication method for Industrial IoT-connected vehicles to provide an efficient and secure vehicle booking and offline authentication. The proposed scheme utilized a smartphone, username, password, and biometric (fingerprint). The combination of AES-ECC is used on the sender and receiver sides. This combination provides secure communication and efficient key management. Furthermore, it gives secure mutual authentication between the vehicle and the cross-server. The proposed scheme comprises five phases, i.e., setup, vehicle registration, server registration, booking, and offline authentication. Figure 3 shows a the general overview of the system's architecture. The notation and descriptions are illustrated in Table 2. Furthermore, the network diagram of the proposed scheme is shown in Figure 4.

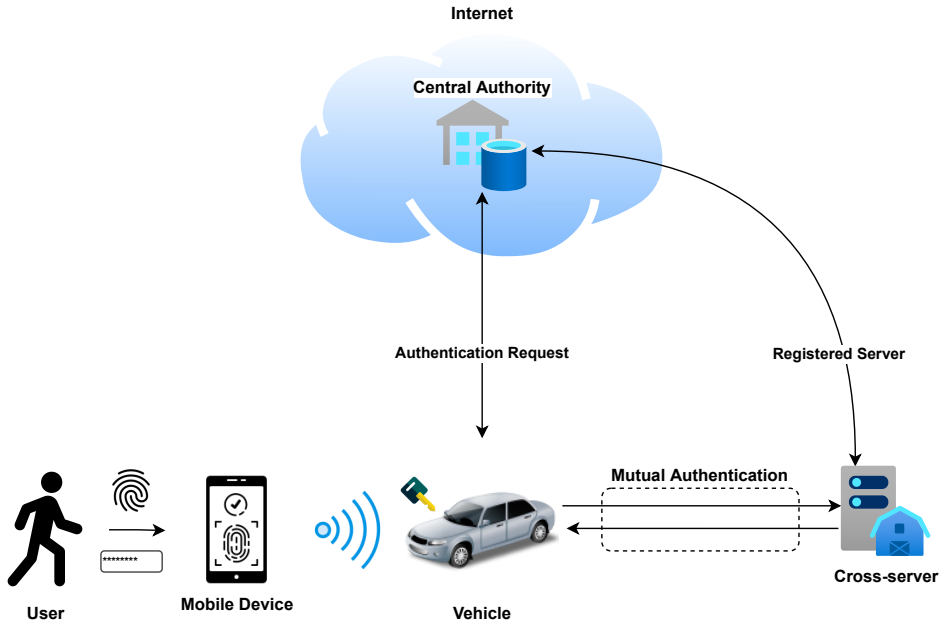

Figure 3. System Architecture.

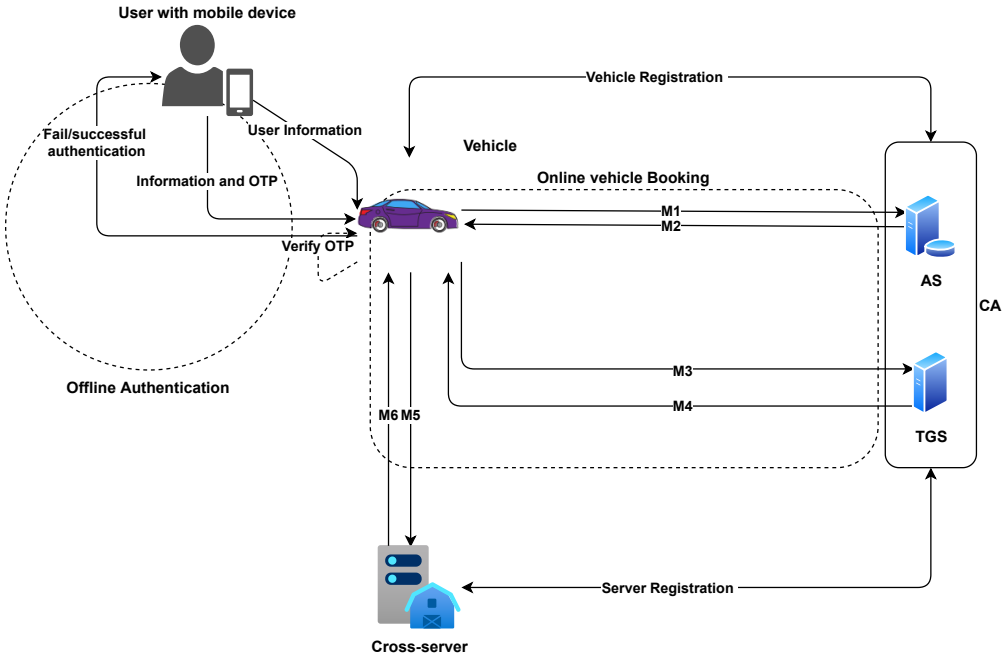

Figure 4. Network diagram of the proposed scheme.

5.1. Setup Phase

In this phase, the trusted authority selects a large prime numbers p and q, and a finite field of elliptic curve E/Fp. Then, the elliptic curve generates a group G with the generator P. The TA then generates the system master key selected randomly, $SMK \in Z_q^*$, and the system public key based on the master key and the prime number $SPK = SMK.p$. The encryption and decryption pair $E./D.$ related to the AES-ECC algorithm are chosen by the TA. Then, it selects a one-way hash function $h() : 0, 1 * ßZ * q$. Later, it computes the corresponding AES public key PK_aes The AES's public key is used to encrypt the transmitted message amongst participating entities. The elliptic curve is used to generate the keys since it has a decent efficiency in key generation. Finally, the TA publishes the public parameters of the system $p, G, SPK, PK_aes, E./D., h(.)$ and key the SMK secretly.

Table 2. Notation.

Notation	Description
p,q	Large prime numbers
TA	Trusted Authority
V_i	Vehicle
AS	Authentication Server
TGS	Ticket Granting Service
CS	Cross-Server
$E./D.$	Encryption/Decryption Pairs
$SMK \in Z_q^*$	System master key
SPK	System public key
PK_{aes}	AES public Key
ID_u	User's Identity
ID_{MD}	Mobile device identity
Pw_u	User Password
Bi_u	User Biometric
ID_V	Vehicle Identity
k_u	User's private key
Pk_u	User Public key
CN	Check number
OTP	One-time Password
ID_{CS}	Cross-server identity
r_1, r_2	Random number
$h(\,) : 0, 1^* \rightarrow Z^*q$	One-Way hash function

5.2. User Registration Phase

To enable the user to be authenticated by the cross-server, the user first needs to register itself into the authentication server AS in the TA. This phase is implied in online mode, where network connectivity is mandatory to complete the user's registration. First, however, the vehicle starts the registration by inserting the smart card into the card reader in the car and selecting the identity, password, and biometric. When the AS receives the message, it checks whether the user exists in the database; if yes, it sends a notification about other information. Otherwise, the AS will start performing the user registration by applying the following steps, as shown in Figure 5:

1. The user inputs the identity ID_u, passwords Pw_u, and imprint biometrics Bi_u. The mobile device picks a pseudonym identity $ID_V \in Z_q^*$. Then, it generates a random number $k_u \in Z_q^*$ as a private vehicle key and calculates the vehicle public key $Pk_u = k_u.p$. Later, it generates a random number r_1 and computes $C_i = ID_V \oplus h(ID_u \oplus Pw_u \oplus r_1)$. The C_i the message is encrypted with AES public key $M_1 : E_{PK_{aes}}\{C_i \| Bi_u\}$ alongside the user biometric and sends M_1 to AS.
2. Upon receiving M_1, the AS decrypts the message $D_{PK_{aes}}\{C_i \| Bi_u\}$ using the AES public key to obtain the user's information.
   The AS then verifies the user's identity and password with the one stored in its database; if it exists, it notifies the user Ui. Otherwise, it computes $V_i = h(ID_u \| d)$ where d is the AS's private key. Then, the mobile device MD generates nonce based on the timestamp and a check number (CN) $nonce = T_1 \| CN$, where CN is a six-digit number for user identification which is a value obtained by calculating the mod nanosecond. Then it encrypts the nonce with $HA = E_{k_{MD}}(nonce)$. Furthermore, it computes $X_i = V_i \oplus h(Bi_u \| C_i \| ID_V)$, and $R_i = h(V_i \| X_i \| C_i)$. The AS calculates $\{V_i, X_i, R_i, p, q, h(.)HA, CN\}$ and stores it in the database. Finally, it sends the parameters to the Ui.
3. After receiving $\{V_i, X_i, R_i, p, q, h(.)HA, CN\}$, it stores the parameters in the memory of the vehicle module.

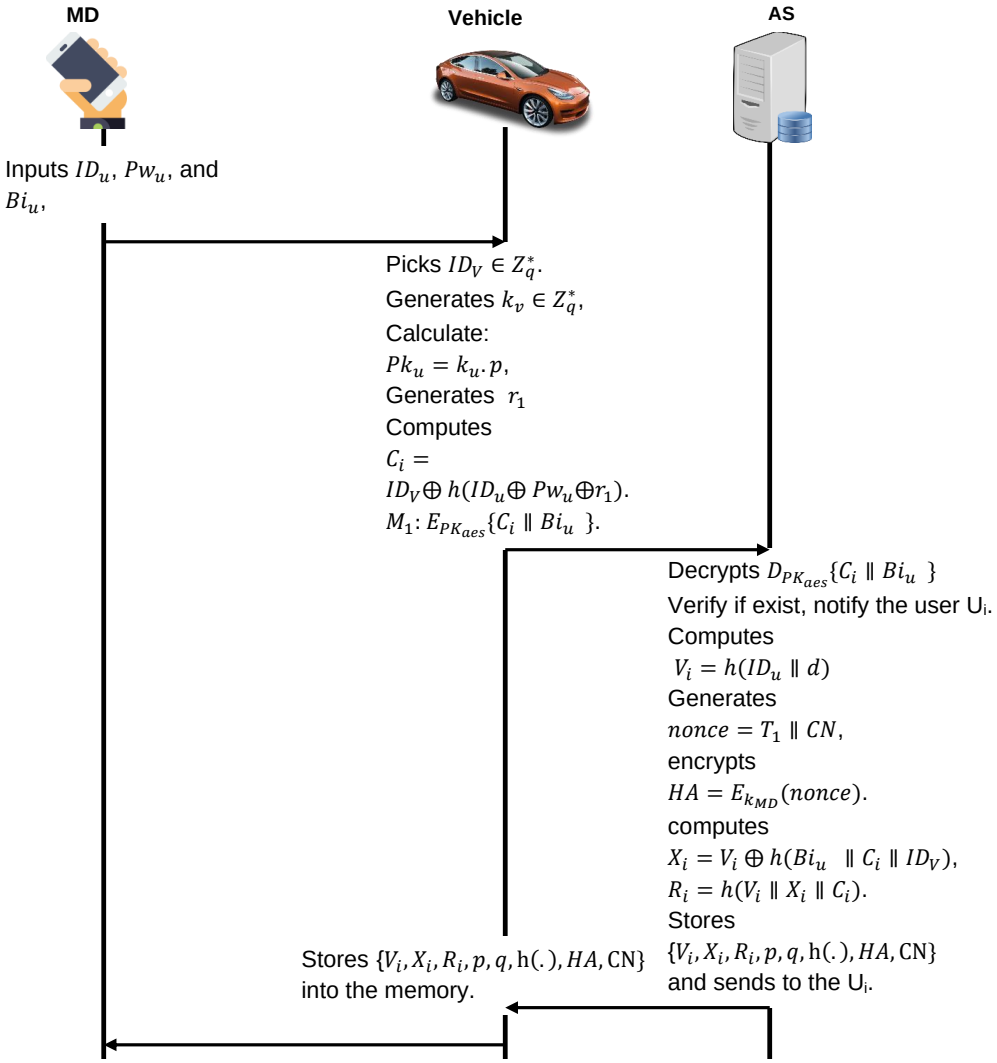

Figure 5. The user registration phase.

5.3. Server Registration Phase

In this phase, the cross-server CS_i register itself into the authentication server in online mode. For server registration, the following steps are shown in Figure 6 and described as follows:

1. The CS_i picks an identity ID_{CS} and select a random number $r_2 \in Z_q^*$ to compute $A_i = h(ID_{CS} \| r_2)$. Then encrypt the message with AES's public key $M_1 : E_{PK_{aes}}\{A_i\}$ and sends M_1 to the AS.
2. The AS receives M_1 and decrypts $D_{PK_{aes}}\{A_i\}$ using the public key to obtain the value A_i, It checks whether the identity exists in the database or not; if so, the AS ask to reapply the registration. Otherwise, it generates a random number $r_3 \in Z_q^*$ and computes a session key shared between the server and the AS $KS_{CS \to AS} = h(ID_{CS} \| r_3)$. Furthermore, the AS will prepare a list that contains all the registered cards in the reader $CR_{list} : \{ID_{CS}, ID_V, V_i, X_i, R_i, r_2\}$ and a secret value S_i to identify the server. Finally, the AS sends the $M_2 : E_{PK_{aes}}\{CS_{list}, S_i\}$ to the server.
3. Upon receiving, the CS_i decrypts the message $D_{PK_{aes}}\{CS_{list}, S_i\}$ using AES public key and obtain the $\{CS_{list}, S_i\}$. Then, it stores the values in its memory for future access.

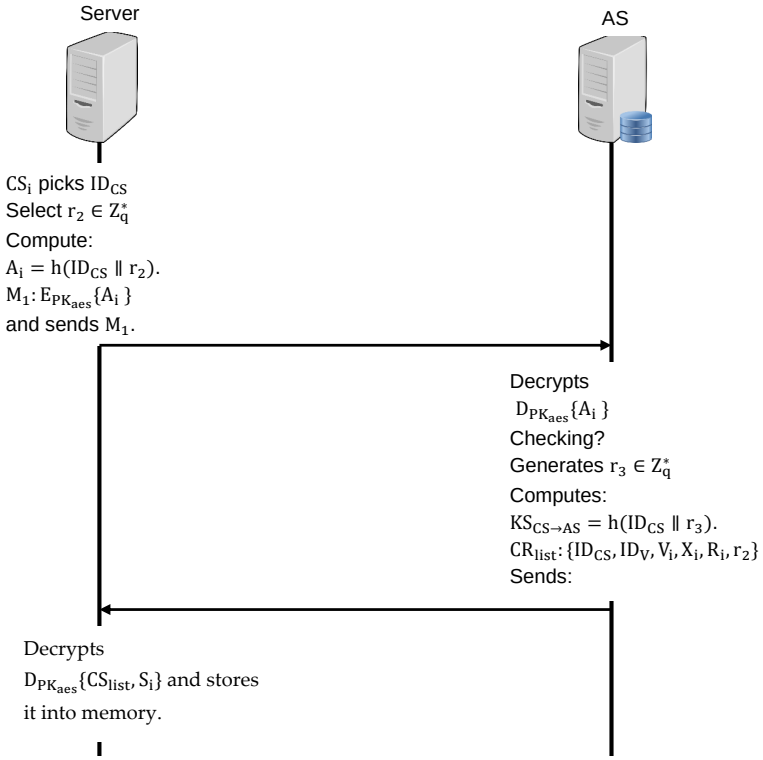

Figure 6. The server registration phase.

5.4. Online Vehicle Booking

In this phase, the user books the vehicle in online mode. The user enters the identity, password, and biometric using their mobile device. To book the vehicle, the user applies the following steps, as described in Figure 7:

1. The user inputs his identity ID_u, passwords Pw_u, and imprint biometrics Bi_u. The MD picks a pseudonym identity $ID_MD \in Z_q^*$. Then, it generates a random number $k_MD \in Z_q^*$ as MD private key and calculates the MD public key $Pk_{MD} = k_{MD}.p$. Later, MD generates a random number r_1 and computes $C_i = ID_V \oplus h(ID_u \oplus Pw_u \oplus r_1)$. The C_i message is encrypted with an AES public key $M_1 : E_{PK_{aes}}\{C_i \| Bi_u\}$ alongside the user's biometric and sends M_1 to the vehicle.
2. Upon receiving M_1, the vehicle decrypts the message $D_{PK_{aes}}\{C_i \| Bi_u\}$ using an AES public key to obtain the user's information. The vehicle then verifies the user's identity and password with the ones stored in its database. If they exist, notify the user's Ui. Later, the vehicle computes 32 bits $TOTP : TOTP = k_v.\triangle T$, where $\triangle T$ is the current time of the vehicle.
   The vehicle computes $C_i = ID_V \oplus h(ID_u \oplus Pw_u)$ and encrypts $M_2 : E_{PK_{aes}}\{C_i, OTP, Bi_u\}$; then sends M_2 to the AS.
3. When the AS receives M_2 it decrypts the message $D_{PK_{aes}}\{C_i \| OTP \| Bi_u\}$ to obtain the values $\{C_i \| OTP \| Bi_u\}$ and verify the identity of the user and vehicle, and checks the freshness of the timestamp $T_1 \neq \triangle T$. If invalid, the session is ended. Furthermore, it verifies the received TOTP by matching with generated TOTP batch by the AS; if there is not a match, the session is ended; otherwise, it causes a random number r_3 and computes a shared key session to enable the vehicle to communicate with the ticket granting service (TGS) $KS_{V \to TGS} = h(ID_v \oplus r_3)$. Then, the AS in trusted authority generates a secret key randomly for the TGS $SK_{TGS} \in Z_q^*$. It later computes the message $M_2 : E_{PK_{aes}}\{ID_u \| ID_V \| T_2 \| KS_{V \to TGS} \| TGS_{TKT}\}$, where the $TGS_{TKT} = E_{SK_{TGS}} ID_u \| ID_V \| T_2 \| S_i$ that can be decrypted by the TGS only. Finally, AS sends M_3 to the vehicle.

4. The vehicle gets the M_2 and decrypts it using the AES public key to obtain $\{ID_u \| ID_V \| T_2 \| KS_{V \to TGS} \| TGS_{TKT}\}$. Then, it forwards the message $M_4 : E_{KS_{V \to TGS}}\{ID_u \| ID_V \| T_2 \| TGS_{TKT}\}$ to the TGS in the trusted authority.
5. The TGS receives the message M_4 and decrypts it to obtain $\{ID_u \| ID_V \| T_2 \| TGS_{TKT}\}$, and then decrypts the ticket $TGS_{TKT} = D_{SK_{TGS}}\{ID_u \| ID_V \| T_2 \| S_i\}$.
The TGS verifies the secret value $S_i \neq S_i'$; if invalid, it ends the session; otherwise, it checks the freshness of the timestamp $T_2 \neq \triangle T$. If not new, it ends the session. Otherwise, it generates a random number r_4 and computes the key session shared between the vehicle and the cross-server $KS_{V \to CS} = h(ID_{TGS} \oplus r_4)$. Then, composes the message $M_4 : E_{KS_{V \to CS}}\{ID_u \| ID_V \| T_3 \| CS_{TKT}\}$, where CS_{TKT} is $CS_{TKT} = E_{PK_{aes}}\{ID_u \| ID_V \| E_{KS_{V \to CS}}\{ID_{CS} \| T_3 \| S_i\}$. Finally, the TGS sends the M_5 to the vehicle.
6. Upon receiving the M_5, the vehicle decrypts the message to obtain the session key and the CS_{TKT}. Then, it decrypts the ticket $D_{PK_{aes}}\{ID_u \| ID_V \| E_{KS_{V \to CS}}\{ID_{CS} \| T_3 \| S_i\}$. It forwards $M_6 : CS_{TKT} = E_{PK_{aes}}\{ID_u \| ID_V \| E_{KS_{V \to CS}}\{ID_{CS} \| T_3 \| S_i\}$ to the cross-server.
7. The cross-server receives the M_6 and decrypts $CS_{TKT} = E_{PK_{aes}}\{ID_u \| ID_V \| E_{KS_{V \to CS}}\{ID_{CS} \| T_3 \| S_i\}$ to obtain the values. Then, it checks the freshness of the timestamp $T_5 \neq \triangle T$. If not fresh, it ends the session; otherwise, the vehicle booking is successfully made.

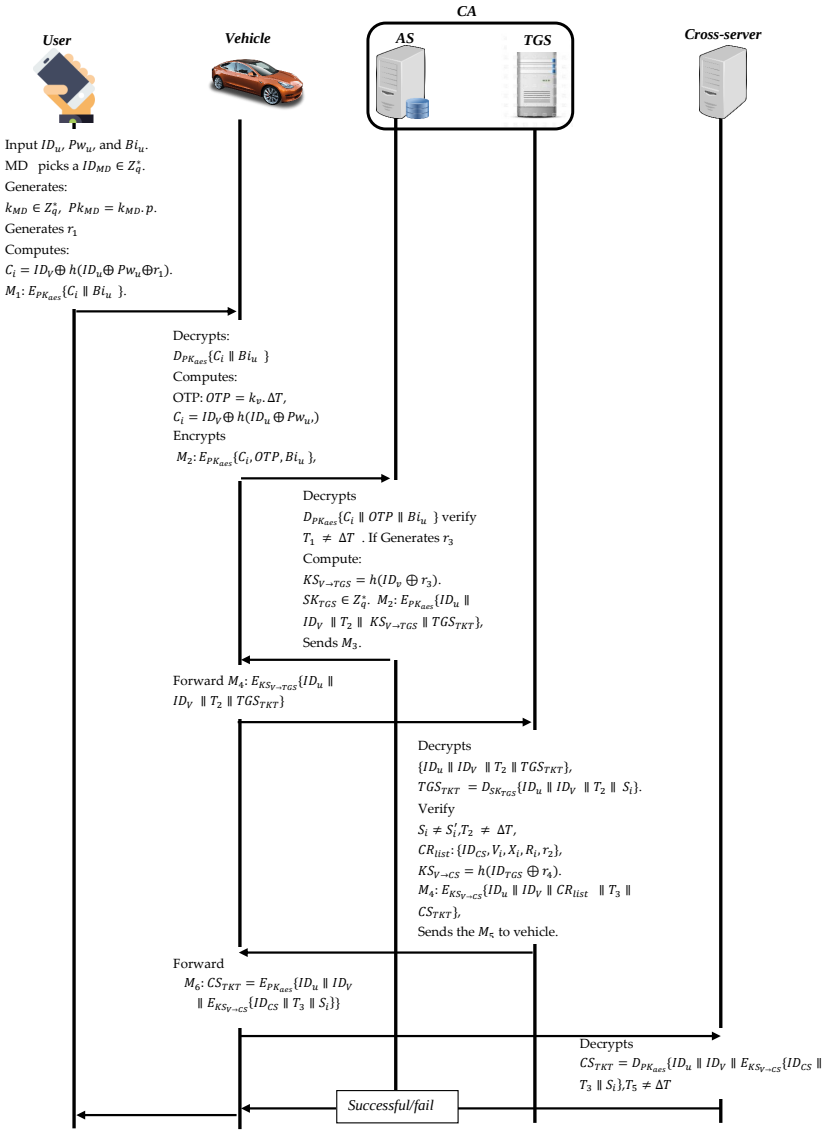

Figure 7. The online vehicle booking phase.

5.5. Offline Authentication

This phase enables the user to authenticate into the offline mode using TOTP [54] when the network connectivity is not available. In Figure 8, the network diagram of the offline authentication is shown. To authenticate the user with the vehicle, the user needs to apply the following steps, as illustrated in Figure 9.

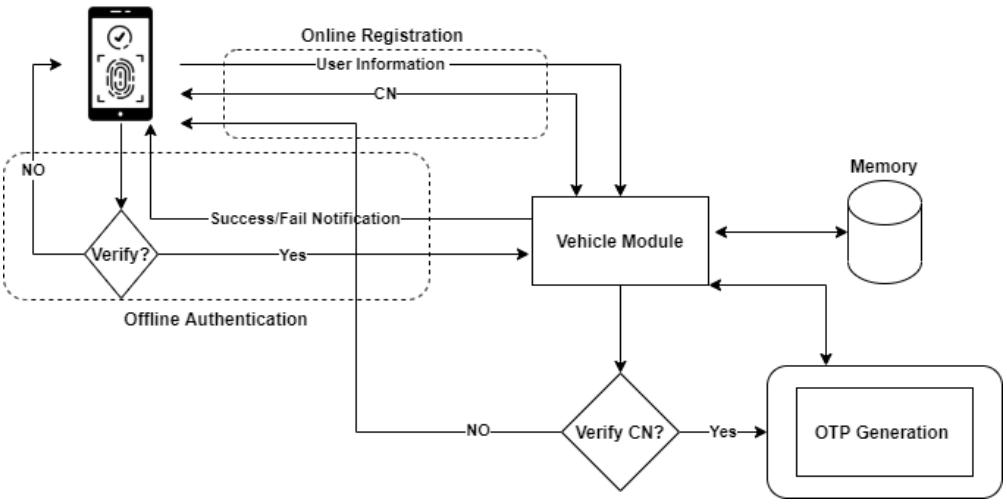

Figure 8. The offline authentication network diagram.

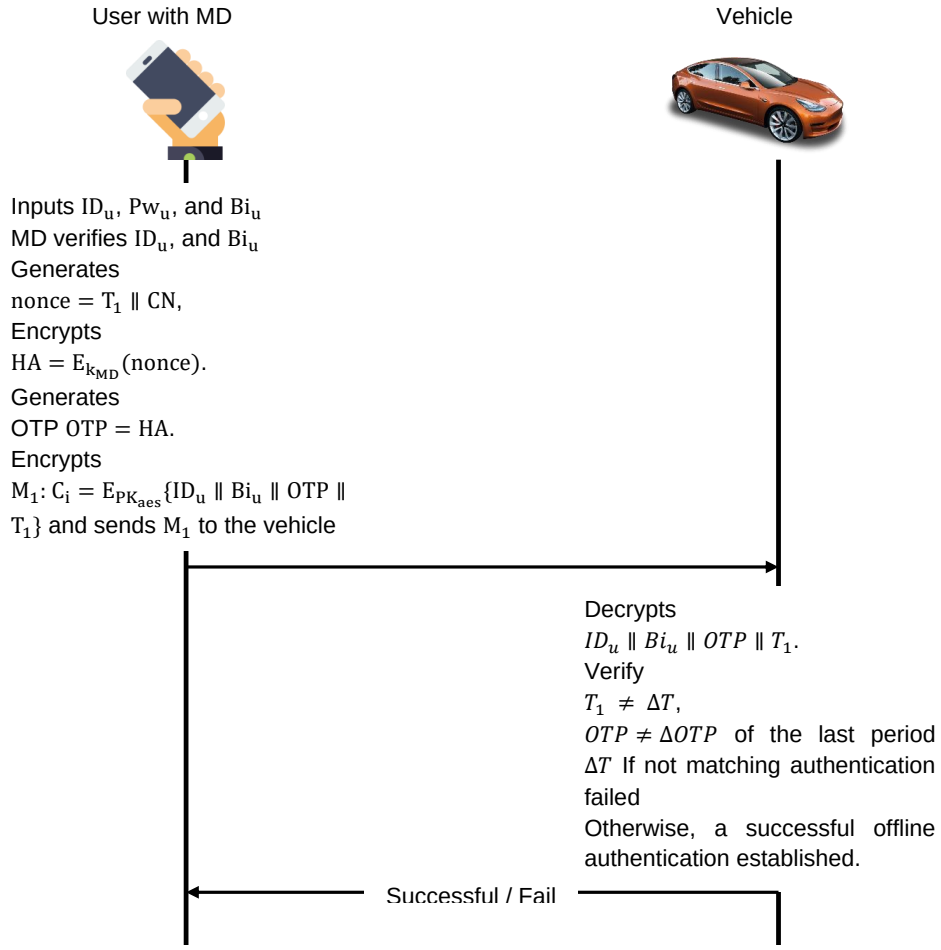

Figure 9. The offline authentication phases.

1. The user inputs their information identity ID_u, passwords Pw_u, and imprint biometrics Bi_u using the mobile device. The mobile device verifies the identity ID_u, and biometrics Bi_u with its database. Then, the mobile device MD generates nonce based on the timestamp and checks number $(CN) nonce = T_1 \| CN$, where CN is a six-digit number for user identification which is a value obtained by calculating the mod in nanoseconds. Then encrypts the nonce with $HA = E_{k_{MD}}(nonce)$. Later, it generates the time one-time password TOTP $OTP = HA$. The mobile device encrypts the message with AES public key $M_1 : C_i = E_{PK_{aes}}\{ID_u \| Bi_u \| OTP \| T_1\}$ and sends M_1 to the vehicle.
2. Upon receiving M_1, the vehicle module decrypts it and obtains $\{ID_u \| Bi_u \| OTP \| T_1\}$. The vehicle verifies the parameters by checking the timestamp $T_1 \neq \triangle T$, and verifies the identity and biometric. The TOTP is confirmed with the generated batch of TOTPs of the last period $\triangle T$ by the vehicle. If the provided TOTP does not match one of the batches and the number or the authentication attempts exceeds ten times, the authentication fails. A failed authentication notifies the user. Otherwise, successful offline authentication is established.

6. Security Analysis

This section discusses the proposed scheme's security review. We first performed an informal and theoretical security review to show that the proposed scheme is secure and functional. Then, a formal security analysis using SVO logic was performed; more details are reported in the following paragraphs.

6.1. Informal Security Analysis

The informal security analysis is shown in this subsection to provide a deep discussion on securing against various attacks. The security properties and functionality are also provided to ensure that the proposed scheme meets the security requirements. Table 3 provides the security feature comparison.

1. Mutual Authentication: The authentication scheme must provide mutual authentication between all the considered entities in the system. In the proposed scheme, the user can communicate with all entities by verifying the timestamp $T_1 \neq \triangle T$. The freshness of the session key $KS_{V \to TGS} = h(ID_v \oplus r_3)$ is checked by the AS and the TGS. Furthermore, the one-time password (OTP) is verified by the server with the generated batch to check the message's validity and the OTP. Therefore, the proposed scheme provides a mutual authentication property.
2. Froward secrecy: In the proposed scheme, the vehicle and the server compute the session key as $KS_{V \to TGS} = h(ID_v \oplus r_3), KS_{V \to CS} = h(ID_{TGS} \oplus r_4)$; hence, the adversary cannot obtain the values because the session key is encrypted with an AES public key and also the ECC key. Furthermore, a new random number is involved in calculating the key session, and the attacker cannot obtain the random value. The user's information is further protected using the one-way hash function $C_i = ID_V \oplus h(ID_u \oplus Pw_u \oplus r_1)$.Thus, the adversary cannot obtain the user's bits; therefore, the proposed scheme provides perfect forward secrecy.
3. Anonymity: Anonymity is essential to protect the user's information in the communication between the entities, since the transmission is done via a public channel. The user's information is calculated $C_i = ID_V \oplus h(ID_u \oplus Pw_u \oplus r_1)$ and encrypted further with AES public key $M_1 : E_{PK_{aes}}\{C_i \| Bi_u\}$. The adversary will not be able to get the user identity, password, and biometric. Furthermore, this information is protected using a one-way hash function as well. Furthermore, the key session $KS_{V \to TGS} = h(ID_v \oplus r_3)$, $KS_{V \to CS} = h(ID_{TGS} \oplus r_4)$, is generated freshly in every communication, and the adversary cannot track the communication between the entities. Assume the adversary could obtain the key session of the current transmission; he/she could not obtain the key session of the next communication. Therefore, the proposed scheme provides anonymity.

4. Confidentiality: The proposed scheme ensures the confidentiality of the message by using a fresh random number $C_i = ID_V \oplus h(ID_u \oplus Pw_u \oplus r_1)$. The message is also encrypted using AES public key $E_{PK_{aes}}\{C_i \| Bi_u\}$ in every communication amongst the entities. In the offline authentication, the mobile device encrypts the value that used to calculate the OTP with device key $HA = E_{k_{MD}}(nonce)$. Furthermore, it encrypts the message with an AES key before transmission $M_1 : C_i = E_{PK_{aes}}\{ID_u \| Bi_u \| OTP \| T_1\}$. Furthermore, the key session is generated independently in each communication. Therefore, the proposed scheme guarantees the confidentiality of the message.
5. Integrity: The integrity of the messages transmitted during the authentication process is guaranteed in the proposed scheme, as the scheme verifies the message by comparing the received values with stored ones in the AS, TGS, and cross-server $T_1 \neq \triangle T$. The verification depends on the timestamp's freshness and the secret values on the server side by calculating them to confirm the message's authenticity. Therefore, the proposed scheme provides message integrity.
6. Key freshness: In the proposed scheme, the shared key session is generated as $KS_{V \to TGS} = h(ID_v \oplus r_3)$, $KS_{V \to CS} = h(ID_{TGS} \oplus r_4)$, independently. The key session is generated freshly in every communication session amongst the vehicle, AS, TGS, and cross-server. Hence, the adversary cannot obtain the key session since it calculates the identity and fresh random number. Assume the adversary obtained the current key session; he/she will not get the session key of the next communication. Therefore, the freshness of the key is provided by the proposed scheme.
7. Offline Authentication: The proposed scheme provides an offline authentication between the mobile user device and the vehicle by providing the vehicle OTP=HA, and $C_i = E_{PK_{aes}}\{ID_u \| Bi_u \| OTP \| T_1\}$. The vehicle checks the timestamp $T_1 \neq \triangle T$, and verifies the identity and biometric. Furthermore, The TOTP is confirmed with the batch of TOTPs generated in the last period $\triangle T$ by the vehicle. If the provided OTP does not match one of the batches, the authentication fails; otherwise, successful offline authentication is established. Therefore, the proposed scheme provides offline authentication between the user and the vehicle.
8. Cross-domain authentication: The vehicle can authenticate to any server registered with a central authority and in a different geographical location. The proposed scheme allows the user to authenticate with the server in the booking phase applied in online mode. The vehicle sends an authentication request to the central authority to get the ability to authenticate with a cross-server. Therefore, the proposed scheme provides cross-domain authentication.
9. Replay attack: The freshness of the messages is guaranteed in each session, since the message is composed of a fresh timestamp T_n. The tickets and the messages $M_2 : E_{PK_{aes}}\{ID_u \| ID_V \| T_2 \| KS_{V \to TGS} \| TGS_{TKT}\}$, $CS_{TKT} = E_{PK_{aes}}\{ID_u \| ID_V \| E_{KS_{V \to CS}}\{ID_{CS} \| T_3 \| S_i\}\}$ include a fresh timestamp. Upon receiving the message, the receiver checks the freshness of the timestamp by comparing it with the current time of the system $T_1, T_2, T_3 \neq \triangle T$. The $\triangle T$ usually is very small to make it difficult for the adversary to replay the captured message within $\triangle T$. The message is further encrypted with a temporary session key make it computationally infeasible for the adversary to modify the composing timestamp. Therefore, the proposed scheme is resilient to replay attacks.
10. Impersonation attack: The adversary who wants to impersonate the user must calculate a valid $C_i = ID_V \oplus h(ID_u \oplus Pw_u \oplus r_1)$in the online booking phase. The values $h(ID_u \oplus Pw_u \oplus r_1)$ are protected using a one-way hash function, and the adversary cannot decipher such values. Additionally, post-transmission is encrypted with the AES key for secure communication and to prevent attackers from capturing the user's information. Therefore, the proposed scheme is resilient to impersonation attacks.
11. Modification attack: In the proposed scheme, the message integrity is preserved using a one-way hash function to protect the user's information. for example, the element $O = hash(t)$ guarantees prevention against modification attacks. Any alteration in

the value O can easily be identified during the comparison and reconstruction of the message at another entity. Furthermore, the messages exchanged amongst entities are encrypted using AES public key, and the communication is retained. Assume the adversary captured the message $M_2 : E_{PK_{aes}}\{ID_u \| ID_V \| T_2 \| KS_{V \to TGS} \| TGS_{TKT}\}$; it is computationally challenging for the adversary to make any changes as the information is encrypted with a temporary session key. Similarly, the exchanged messages are ciphered to prevent any modification. Therefore, the proposed scheme withstands the modification attack.

12. Man-in the middle attack: In this attack, the adversary tries to modify the captured message in a way where the destination cannot differentiate the modified message from the original message. Assume the adversary applies an MiTM attack between the vehicle and the AS by capturing and modifying the message $M_2 : E_{PK_{aes}}\{C_i, OTP, Bi_u\}$, or the message between the vehicle and the TGS $M_4 : E_{KS_{V \to TGS}}\{ID_u \| ID_V \| T_2 \| TGS_{TKT}\}$. The message's construction is computationally hard for the adversary, as the message is double encrypted with both the fresh session key and an AES public key. Furthermore, the tickets, such as $CS_{TKT} = E_{PK_{aes}}\{ID_u \| ID_V \| E_{KS_{V \to CS}}\{ID_{CS} \| T_3 \| S_i\}$ are encrypted, and each contains a ciphered timestamp and secret value that will be validated later by their respective destinations. Therefore, the proposed scheme is protected from a man-in-the-middle attack.
13. Server spoofing attack: This attack tries to spoof a server by replaying an old authentication message $M_{2_i^{old}} : E_{PK_{aes}} ID_u \| ID_V \| T_2 \| KS_{V \to TGS} \| TGS_{TKT}$. This attempt fails, since the user uses a new and different random number, and a fresh session key is used as well, which means the session key is used differently in each session. The session key of the current communication is different from those of the last and next sessions. Therefore, the proposed scheme is resilient to a server spoofing attack.
14. Privileged insider attack: A privileged attack can allow access to a user's information. Assume the adversary has the registration information of the user identity ID_u, Pw_u, and Bi_u; he/she cannot guess the information as the information is protected using a one-way hash function $C_i = ID_V \oplus h(ID_u \oplus Pw_u \oplus r_1)$ and composed with a random number. Furthermore, the information is ciphered before transmission. Therefore, the proposed scheme withstands a privileged insider attack.
15. Denial of service (DoS) attack: A DoS attack makes the server unavailable. In the proposed scheme, the timestamp T_n is used to check the freshness of the message. In the booking phase, if the user and central authority exchanged the messages $M_2 : E_{PK_{aes}}\{ID_u \| ID_V \| T_2 \| KS_{V \to TGS} \| TGS_{TKT}\}$, $CS_{TKT} = E_{PK_{aes}}\{ID_u \| ID_V \| E_{KS_{V \to CS}}\{ID_{CS} \| T_3 \| S_i\}$, the server checks the timestamp against the current timestamp $T_1, T_2, T_3 \neq \triangle T$; if not fresh, the server rejects the message. Furthermore, the message also included a secret value S_i. The server will check that for the validity of the message. As a result, the proposed scheme is secure against the DoS attacks.
16. Offline guessing attack: With the assistance of the side-channel attacks such as SPA and DPA, the adversary cannot obtain $C_i = ID_V \oplus h(ID_u \oplus Pw_u \oplus r_1)$ because the user's information is protected using the one-way hash function. Even if the adversary obtains $ID_u \| ID_V \| T_2 \| KS_{V \to TGS} \| TGS_{TKT}$, he/she cannot decipher the ticket in the offline environment and encrypted using the temporary session key. Therefore, the proposed scheme withstands the offline guessing attack.

Table 3. A comparison of security features.

	SM-AKA [40]	HOOSC [39]	CP-VBP [20]	RSEAP [18]	Proposed Scheme
Mutual Authentication	✓	✓	✓	✓	✓
Froward secrecy	✓	✓	x	✓	✓
Anonymity	✓	x	x	✓	✓
Confidentiality	x	x	✓	✓	✓
Integrity	x	x	x	✓	✓
Key freshness	x	✓	✓	x	✓
Offline Authentication	✓	✓	x	x	✓
Cross-domain authentication	x	x	x	x	✓
Replay attack	✓	✓	✓	✓	✓
Impersonation attack	✓	x	✓	✓	✓
Modification attack	x	✓	✓	✓	✓
Man-in the middle attack	✓	✓	✓	✓	✓
Server spoofing attack	x	x	x	x	✓
Privileged insider attack	✓	✓	x	✓	✓
Denial of service (DoS) attack	✓	✓	x	✓	✓
Offline guessing attack	x	✓	✓	✓	✓

6.2. Syverson and Van Oorschot (Svo) Logic

A growing number of researchers are turning to systematic analysis to assess their protocols and schemes' security. Syverson and Van Oorschot's (SVO) logic [45], as a significant BAN-like logic, possesses the advantages of BAN logic, GNY logic, and AT logic. Furthermore, SVO logic redefines certain standard semantic principles and has fundamental inference rules or axioms. SVO logic is now a commonly used formal analysis technique. However, we provide formal security proof of the proposed scheme using SVO logic in this subsection. Table 4 gives the notation that is used in SVO logic and relevant descriptions.

Table 4. The SVO logic notation and the corresponding descriptions.

Notation	Description
$\dashv \varphi$	φ is a theorem
$PK_{\sigma}(P,K)$	K is the public signature verification key for P
$PK_{\delta}(P,K)$	K is the public key-agreement key for P
$SV(X,K,Y)$	K can verify if X is Y's signature
$Fresh(X)$	X is fresh
XK	The ciphertext encrypted by K
$[X]K$	The message signed by K

Initial Rules:The SVO logic has two main inference rules:

1. The Separation rule Modus Ponens (MP) from φ and $\varphi \supset \psi \Rightarrow \psi$.
2. The necessity of rules Necessitation (Nec) from $\dashv \varphi \Rightarrow$ believing φ.

SVO axiom:For any subject P and Q, the sum of the formulas φ and ψ have the following axiom schemata's:

1. Believes:
   Ax.1: P believes $\varphi \wedge P$ believes $(\varphi \supset \psi) \supset P$ believes ψ.
   Ax.2: P believes $\varphi \supset P$ believes $(P \; believes \; \varphi)$.
2. Source Association:
   Ax.3: Sharedkey $(K,P,Q) \wedge R$ received $\{X^{Q}\}_{K} \supset Q$ said $X \wedge Q$ sees K.
   Ax.4: $PKs_{Q}\ (Q,K) \wedge R$ received $X \wedge SV(X,K,Y) \supset Q$ said Y.

3. Key agreement:
   Ax.5: $PK_\sigma(P, K_P) \wedge (Q, K_Q) \supset$ SharedKey $(F(K_P, K_Q), P, Q)$.
4. Receiving:
   Ax.6: P received $(X_1, \ldots, X_n) \supset P$ receives X_i.
   AX.7: P received $\{X\}_K \wedge P$ sees $K^{-1} \supset P$ receives X.
5. Seeing:
   Ax.8: P received $X \supset P$ sees X.
   Ax.9: P sees $(X_1, \ldots, X_n) \supset P$ sees X_i.
   Ax.10: P sees $X_1 \wedge \cdots \wedge P$ sees $F(X_1, \ldots, X_n)$.
6. Comprehending:
   Ax.11: P believes $(P \textit{ sees } F(X)) \supset P$ believes (P sees X).
   Ax.12: (P received $F(X) \wedge P$ believes P sees X) $\supset$ P believes P received $F(X)$.
7. Saying:
   Ax.13: P said $(X_1, \ldots, X_n) \supset P$ said $X_i \wedge P$ sees X_i.
   Ax14: P says $(X_1, \ldots, X_n) \supset P$ says $X_i \wedge P$ said $(X_1, \ldots, X_n)$.
8. Jurisdiction:
   Ax.15: (P controls $\wedge P$ says φ) $\supset \psi$.
9. Freshness:
   Ax.16: $fresh(X_i) \supset fresh(X_1, \ldots, X_n)$.
   Ax.17: $fresh(X_1, \ldots, X_n) \supset (F(X_1, \ldots, X_n))$.
10. Nonce-Verification:
    Ax.18: $fresh(X) \wedge P \textit{ said } X \supset P \textit{ says } X$.
11. Symmetric goodness of shared keys:
    Ax.19: $SharedKey(K, P, Q) \equiv SharedKey(K, Q, P)$.
12. Having:
    Ax.20: P has $K \supset P$ sees K.

Goals: In the following SVO logic, the goals are given according to the security requirements of the proposed scheme to prove the security of the scheme:

1. Goal.1: Vehicle believes User says (ID_V, ID_u, Pw_u, r_1).
2. Goal.2: AS believes vehicle says (C_i, OTP, Bi_u).
3. Goal.3: Vehicle believes AS says $(ID_u, ID_V, T_2, KS_{V \to TGS}, TGS_{TKT})$.
   TGS believes vehicle says $(ID_u, ID_V, T_2, TGS_{TKT})$.
4. Goal.4: Vehicle believes TGS says $(ID_u, ID_V, T_3, CS_{TKT})$.
   CS believes vehicle says (ID_{CS}, T_3, S_i).
5. Goal.5: Vehicle Believes AS says (T_2). CS believes TGS says (T_3).
6. Goal.6: Vehicle believes sharedkey $(KS_{V \to TGS}, Vehicle, TGS)$. CS believes sharedkey $(KS_{V \to CS}, Vehicle, CS)$.
7. Goal.7: Vehicle believes fresh $(KS_{V \to TGS})$. CS believes fresh $(KS_{V \to CS})$.

Assumptions:
AS.1: Vehicle Believes fresh (T_2)
CS believes fresh (T_3)
AS.2: Vehicle believes vehicle received $((ID_u, ID_V, T_2, KS_{V \to TGS}, TGS_{TKT}) \supset PK_\delta(AS, r_3P))$
AS believes AS received $(([C_i, OTP, Bi_u]_{PK_{aes}}) \supset PK_\delta(Vehicle, r_1P))$
AS.3: TGS believes TGS received $(([[ID_u, ID_V, T_2, TGS_{TKT}]_{KS_{V \to TGS}}) \supset PK_\delta(Vehicle, S_iP))$
Vehicle believes vehicle received $([ID_u, ID_V, T_3, CS_{TKT}]_{KS_{V \to TGS}}) \supset PK_\delta(TGS, r_3P))$
CS believes CS received $([ID_u, ID_V]_{PK_{aes}}), [ID_{CS}, T_3, S_i]_{KS_{V \to CS}})) \supset PK_\delta(Vehicle, r_3P))$
AS.4: Vehicle believes vehicle received $[T_2]_{KS_{V \to TGS}}$
TGS believes TGS received $[T_2]_{KS_{V \to TGS}}$
CS believes CS received $[T_3]_{KS_{V \to CS}}$
AS.5: Vehicle believes $PK_\delta(AS, r_3P)$
AS believes $PK_\delta(Vehicle, r_1P)$
TGS believes $PK_\delta(Vehicle, S_iP)$
CS believes $PK_\delta(Vehicle, r_3P)$

AS.6: Vehicle believes SV $([ID_u, ID_V, T_2, KS_{V\to TGS}, TGS_{TKT}]PK_{aes}, SK_{TGS}, (ID_u, ID_V, T_2, S_i))$
AS believes SV $([C_i, OTP, Bi_u]PK_{aes}, (ID_V, ID_u, Pw_u))$
TGS believes SV $([[ID_u, ID_V, T_2, TGS_{TKT}]KS_{V\to TGS}, SK_{TGS}, (ID_u, ID_V, T_2, S_i))$
CS believes SV $(ID_u, ID_V, PK_{aes}, KS_{V\to CS}, (ID_{CS}, T_3, S_i))$
AS.7: Vehicle believes ((AS says $(ID_u, ID_V, T_2, KS_{V\to TGS}, TGS_{TKT}) \supset PK_\delta(AS, r_2P))$
TGS believes ((vehicle says $(ID_u, ID_V, T_2, KS_{V\to TGS}, TGS_{TKT}) \supset PK_\delta(vehicle, r_2P))$
CS believes ((vehicle says $(ID_u, ID_V, ID_{CS}, T_3, S_i, CS_{TKT}) \supset PK_\delta(TGS, r_3P))$
AS.8: Vehicle believes $PK_\delta(vehicle, r_1P)$
AS believes $PK_\delta(AS, r_2P)$
TGS believes $PK_\delta(TGS, r_3P)$
CS believes $PK_\delta(CS, r_3P)$
AS.9: Vehicle believes (vehicle sees $PK_\delta(vehicle, r_1P))$
AS beliefs (AS sees $PK_\delta(AS, r_2P))$
TGS believes $(TGS\ sees\ PK_\delta(TGS, r_3P))$
CS believes $(CS\ sees\ PK_\delta(CS, r_3P))$
AS.10: $\neg$ (vehicle said T_1PK_{aes}
$\neg$ (AS said $T_2KS_{V\to TGS}$
$\neg$ (TGS said $T_3KS_{V\to CS}$)
AS.11: Vehicle believes fresh (T_2)
CS believes fresh (T_3)

Security Proof:

From AS.2, AS.5, AS.6, Ax.4, we can get:
S1: Vehicle believes AS said $ID_u, ID_V, T_2, KS_{V\to TGS}, TGS_{TKT})$
TGS believes the vehicle said $(ID_u, ID_V, T_2, TGS_{TKT})$
Vehicle believes TGS said $(ID_u, ID_V, ID_{CS}, T_3, CS_{TKT})$
CS believes vehicle said $(ID_u, ID_V, ID_{CS}, T_3, S_i, CS_{TKT})$
From S1, AS.1, AS.2, Ax.19, we get:
S2: Vehicle believes User says (ID_V, ID_u, Pw_u, r_1)
AS believes vehicle says (C_i, OTP, Bi_u)
The Goal 1., and Goal 2. are proved.
From S2, AS.5, Ax.1, and Necessitation, we can get:
S3: Vehicle believes $PK_\delta(AS, r_3P)$
AS believes $PK_\delta(Vehicle, r_1P)$
TGS believes $PK_\delta(Vehicle, S_iP)$
CS believes $PK_\delta(Vehicle, r_3P)$
From S3, AS.8, Ax.5, we can get:
S4: Vehicle believes sharedkey $(KS_{V\to TGS}, Vehicle, TGS)$
CS believes sharedkey $(KS_{V\to CS}, Vehicle, CS)$
Where $KS_{V\to TGS} = h(ID_v \oplus r_3)$,
$KS_{V\to CS} = h(ID_{TGS} \oplus r_4)$
From AS.2, Ax.1, Ax.8, we can obtain:
S5: Vehicle believes (vehicle sees $PK_\delta(vehicle, r_1P))$
AS believes $(AS\ sees\ PK_\delta(AS, r_2P))$
TGS believes $(TGS\ sees\ PK_\delta(TGS, r_3P))$
CS believes $(CS\ sees\ PK_\delta(CS, r_3P))$
From S5, AS.9, Ax.5, we can obtain:
S6: Vehicle believes vehicle sees sharedkey $(KS_{V\to TGS}, Vehicle, TGS)$
CS believes CS sees sharedkey $(KS_{V\to CS}), Vehicle, CS)$
Where $KS_{V\to TGS} = h(ID_v \oplus r_3)$,
$KS_{V\to CS} = h(ID_{TGS} \oplus r_4)$
From S4, S6, the definition of Sharedkey $(K-, P, Q)$, we can get:
S7: Vehicle believes sharedkey $(KS_{V\to TGS}, Vehicle, TGS)$
AS believes sharedkey $(KS_{V\to CS}, Vehicle, CS)$

Thus, Goal.6 is proved.
From AS.7, AS.2, S1, Ax.6, Ax.13, Ax.14, we can obtain:
S8: Vehicle believes ((AS said $(ID_u, ID_V, T_2, KS_{V \to TGS}, TGS_{TKT})$)
TGS believes ((vehicle said $(ID_u, ID_V, T_2, KS_{V \to TGS}, TGS_{TKT})$)
CS believes ((vehicle said $(ID_u, ID_V, ID_{CS}, T_3, S_i, CS_{TKT})$)
Thus, Goal 3, and Goal 4., are proved.
From AS.1, AS.2, S4, Ax.16,Ax.17, we can get:
S9: Vehicle believes fresh $(KS_{V \to TGS})$
CS believes fresh $(KS_{V \to CS})$
Goal 7., proved.
From AS.3, S4, Ax.3, we can obtain:
S10: vehicle believes AS said fresh T_2
TGS believes AS said T_2
CS believes TGS said T_3
From S10, AS.11, and Ax.19, we can get:
S11: vehicle believes AS says T_2
TGS believes AS says T_2
CS believes TGS says T_3
Thus, Goal 5. proved.

According to the formal descriptions of the security proof presented in this subsection, the proposed authentication scheme is secure.

7. The Avispa Simulation

AVISPA offers a broad range of modeling applications for cryptographic protocol analysis, verification, and validation. It describes security protocols using the High-Level Protocol Specification Language (HLPSL) [55]. HLPSL is a high-level, modular language requiring various attacker models, cryptographic primitives, and complex security properties. The protocol was first translated into intermediate form (IF) using the HLPSL2IF translator, and then IF was used as input to four different back-ends, as shown in Figure 10, which included On-the-fly Model-Checker (OFMC), CL-based Attack Searcher (CL-AtSe), SAT-based Model-Checker (SATMC), and Tree-Automata-based Protocol-Analyzer (TA4SP). The transmission channel was also thought to be under the influence of the Dolev-Yao attacker. A few automated security validation tools, such as ProVerif and Scyther tools, were utilized to verify the security of the protocol. Using these tools, it is easy to know what flaws such protocols suffer from. Although verification using these tools does not ensure that the protocols once verified by these tools are flawless, they still provide a means to know many of the flaws easily. ProVerif verified specifications of protocols in the symbolic model, which could also be a limitation, since the symbolic model abstracts away the details of cryptographic operations, and specifications do not consider all implementation details [55]. Unlike other studies, we used the widely used AVISPA formal verification tool because the AVISPA tool is efficient at verifying and falsifying security protocols. This tool can be used for the final analysis of the cryptographic protocol, as it allows one to detect atypical errors according to the established requirements. There were two protocols designed, online booking and offline authentication; hence, the formal verification of the online booking and the offline authentication is discussed further in the following paragraphs.

7.1. Specifying the Online Booking in Hlpsl

The online booking phase's implementation, including the vehicle registration phase, login, and authentication phase, is provided in this subsection. Our simulation had four main roles: vehicle (Vi), authentication server (AS), ticket granting service (TGS), and cross-server (CS), respectively. However, the vehicle's specifications in HLPSL language are shown in Figure 11. The vehicle Vi received the start signal and changed its state from 0 to 1, and the sent the registration request message (V.TGS.cLifetime_1. Bio'. Ci'. N1') to the AS securely using the /\ Snd () operation. In the login phase, the vehicle then generated (OTP':

= Kv.current_time), and sent (CS.cLifetime_2.Uid'.Bio'.TGSid'.TkT1'.{V.OTP'}_K_UG') using the /\ Snd() operation. The declaration /\ witness (V, TGS, t1, OTP') expressed the TGS acceptance of the generated OTP by vehicle. The declaration /\ wrequest (V, AS, k_cg1, K_UG') indicates that the vehicle requested the AS to check the validity of the value K_UG'.

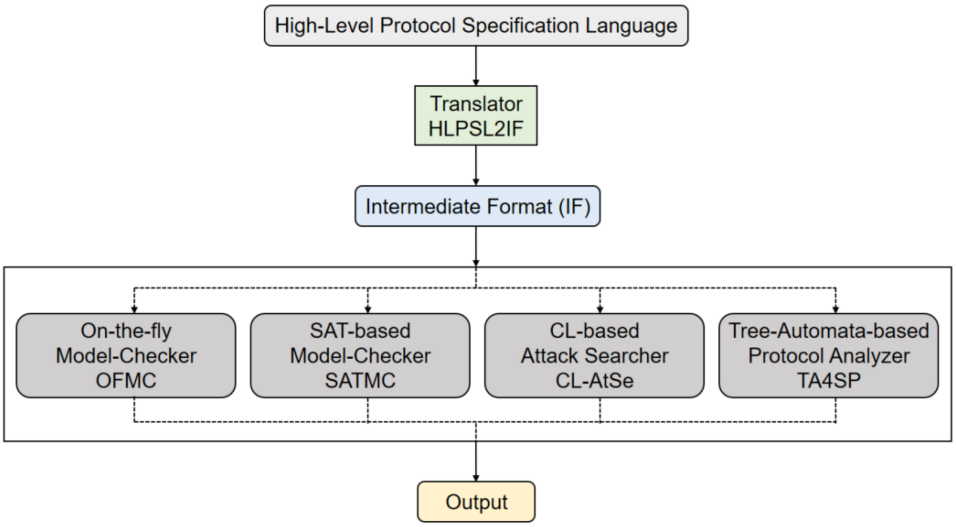

Figure 10. The typical AVISPA architecture.

Later, the vehicle decelerated the /\ secret (K_UG', sec_c_K_UG, {AS,V,TGS}) to indicate that the value K_UG' is known to the agents {AS,V,TGS} and it is confidential. Later, the vehicle received message (V. TkT2'. {CS.K_US'.Ts2'.Tse2'}_K_UG) from the TGS using the operation /\ Rcv(). Likewise, the vehicle sent message (TkT2'.{V.T2'.Uid'.Vid'.TkT2. TGSid'.CSid'}_K_US') to the cross-server using /\ Snd(). Hence, /\ witness (V, CS, t2b, T2') declared for a (weak) authentication property of V by CS on T2, that agent V is the witness for the information T2. /\ wrequest (V, TGS, k_cs1, K_US') declared that the vehicle requested the TGS to check K_US'. The declaration /\ secret(K_US', sec_c_K_US,{TGS,V,CS}) shows that the value K_US' was kept secret from the agent's TGS, V, and CS.

The role of the authentication server is shown Figure 12. The AS received the message (V.TGS.Lifetime_1'.N1'.OTP'.Ci') and changed its state from 0 to 1, which was maintained by the variable state, and then sent the M3':= ({Uid'.Vid'.Ts2'.TGS_tkt'}_Pk_aes) securely to the vehicle. It also generated ticket TGS_tkt':= ({Uid'.Vid'.Ts2'.K_v_tgs'}_SK_tgs) encrypted using the TGS secret key. The declaration /\ witness(AS,V,k_cg1,K_v_tgs') shows that the shared key K_v_tgs was sent to the vehicle by the AS. Likewise, /\ witness(AS,TGS,k_cg2, K_v_tgs') indicates that the TGS believed that the AS generated a fresh K_v_tgs and the AS witnessed it. The declaration /\ secret(K_v_tgs',sec_a_K_UG,{AS, V,TGS}) illustrates that the K_v_tgs' was shared securely and remained secret to the AS, V, and TGS.

Figure 13 shows the role of the ticket granting service (TGS) in HLPSL. The TGS started by receiving (CS.Lifetime_2'.N2'.{V.TGS.K_UG'.Ts'.Tse'}_K_AG.V.T'} _K_UG') using the Rcv() operation. Later, the TGS generated the key session / K_v_cs':= H(TGSid,Ri') and the ticket for CS /\ CS_tkt':= (Uid'.Vid'.CSid'.K_v_cs'.Ts3'_Pk_aes). The TGS sent the message (V.{V.CS.K_US'.Ts2'.Tse2'}_K_GS.{CS.K_US'.Ts2'.Tse2'. N2'}_K_UG') to the vehicle. The declaration /\ wrequest(TGS,V,t1,T') indicates the vehicle checked T' that was generated by TGS, where /\ wrequest(TGS,AS,k_cg2,K_UG') shows that the AS requested the AS to validate the value K_UG'. Furthermore, the declaration /\ witness(TGS,V,k_cs1,K_US') illustrates that V witnessed the K_US' generated by the TGS, where /\ witness(TGS,CS,k_cs2, K_US') declares that the CS witnessed the K_US' that was generated by the TGS. The declaration /\ secret(K_UG',sec_g_K_UG,{AS,V,TGS} indicates that key K_UG' was shared secretly amongst AS, V, and TGS; the declaration /\ secret(K_US',sec_g_K_US,{TGS,V,CS}) shows that the key K_US' was kept secretly amongst TGS, V, and CS.

```
%Vehicle
role vehicle (V, AS, TGS, CS : agent,
                Snd, Rcv   : channel (dy),

                K_UA       : symmetric_key)
played_by V
def=
  local St   : nat,
      K_UG, K_US : symmetric_key,
      T, T2: text,
      H:hash_func,
      Ts, Tse, Ts2, Tse2    : text,
      TkT1, TkT2 : {agent.agent.symmetric_key.text.text}_symmetric_key,
      N1, N2 ,Bio,UBi,Ri,Uid,Upw,Mi,Ai,Fi,Ni,TGSid,CSid,Vid,Ci,OTP,Kv,
      M1,M2,M3,M4,M5,M6 : text
const sec_c_K_UG, sec_c_K_US : protocol_id,
        cLifetime_1, cLifetime_2, current_time: text
  init  St := 0
  transition
%%% vehicle registeration Phase
   1. St = 0  /\ Rcv(start) =|>
      St':= 1 /\ N1' := new()
              /\ Ri':=new()
              /\ Bio':= new()
              /\ Ci':= Vid.xor(H(Uid,Upw,Ri'))
              /\ Snd(V.TGS.cLifetime_1.Bio'.Ci'.N1')
   2. St = 1  /\ Rcv(V.TkT1'.{TGS.K_UG'.Ts'.Tse'.N1.Ri}_K_UA) =|>
%%% online booking phase
      St':= 2
              /\ Mi':= new()
              /\TGSid':=new()
              /\ Bio':= new()
              /\ Ri':= new()
              /\ Uid':=new()
              /\ OTP':= Kv.current_time
              /\ Snd(CS.cLifetime_2.Uid'.Bio'.TGSid'.TkT1'.{V.OTP'}_K_UG')
              /\ witness(V,TGS,t1,OTP')
              /\ wrequest(V,AS,k_cg1,K_UG')
              /\ secret(K_UG',sec_c_K_UG,{AS,V,TGS})
   3. St = 2  /\ Rcv(V.TkT2'.{CS.K_US'.Ts2'.Tse2'}_K_UG)  =|>
      St':= 3 /\ T2' := new()
              /\ TGSid':=new()
              /\ CSid':=new()
              /\ Uid':=new()
              /\ Vid':= new()
              /\ M4':= ({Uid.Vid.T2.TkT1}_K_UG)
              /\ Snd(TkT2'.{V.T2'.Uid'.Vid'.TkT2.TGSid'.CSid'}_K_US')
              /\ witness(V,CS,t2b,T2')
              /\ wrequest(V,TGS,k_cs1,K_US')
              /\ secret(K_US',sec_c_K_US,{TGS,V,CS})
   4. St = 3  /\ Rcv({T2}_K_US) =|>
      St':= 4 /\ wrequest(V,CS,t2a,T2)

end role
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Figure 11. The vehicle's role in HLPSL.

```
% Authentication Server
role k_AS (AS, V, TGS : agent,
                 Snd, Rcv   : channel (dy),
                 K_UA, K_AG : symmetric_key)
played_by AS
def=
  local St              : nat,
        K_UG            : symmetric_key,

        N1, Lifetime_1,Ai,Sk,Uid,Bio,Fi,Ri,Mi,Ni,Xi,Vid,Ci,
        OTP,Kv,M1,M2,M3,M4,M5,M6, K_v_tgs,SK_tgs, Pk_aes,TGS_tkt : text,
        H:hash_func,
        Ts, Tse,Ts2 : text
  const k_cg1, k_cg2 : protocol_id,
        sec_a_K_UG : protocol_id
  init  St := 0
  transition
   1. St = 0  /\ Rcv(V.TGS.Lifetime_1'.N1'.OTP'.Ci') =|>
      St':= 1 /\ Ts' := new()
              /\ Tse' := new()
              /\ K_UG' := new()
              /\ Sk':= new()
              /\ Ts2':= new()
              /\ Vid':= new()
              /\ Ri':=new()
               /\ Bio':=new()
               /\ Uid':= new()
               /\ K_v_tgs':= H(Vid,Ri')
               /\ SK_tgs':= new()
               /\ M3':= ({Uid'.Vid'.Ts2'.TGS_tkt'}_Pk_aes)
               /\ TGS_tkt':= ({Uid'.Vid'.Ts2'.K_v_tgs'}_SK_tgs)
              /\ Snd(V.{V.TGS.K_v_tgs'.Ts'.M3'.TGS_tkt'}_K_AG.
                       {TGS.K_v_tgs'.Ts'.M3'}_K_UA)
              /\ witness(AS,V,k_cg1,K_v_tgs')
              /\ witness(AS,TGS,k_cg2,K_v_tgs')
              /\ secret(K_v_tgs',sec_a_K_UG,{AS,V,TGS})
end role
```

Figure 12. The AS's role in HLPSL.

```
% Ticket Granting Server
role k_TGS (TGS, AS, CS, V  : agent,
                Snd, Rcv    : channel (dy),
                K_AG, K_GS  : symmetric_key)
played_by TGS
def=

  local St                                : nat,
        K_UG                              : symmetric_key,
        K_US                              : symmetric_key,
        Lifetime_2, Ts, Tse, T, N2,Ai,Sk,Uid,CSid,Bio,Fi,Ri,Mi,Ni,Xi,
        Vid,TGSid,Ci,OTP,Kv,M1,M2,M3,M4,M5,M6, K_v_cs,SK_tgs, Pk_aes,
        CS_tkt  : text,
        H:hash_func,
        Ts2, Ts3,Tse2                  : text
  const t1, k_cs1, k_cs2 : protocol_id,
        sec_g_K_UG, sec_g_K_US : protocol_id
  init  St := 0
  transition
   1. St = 0  /\
       Rcv(CS.Lifetime_2'.N2'.{V.TGS.K_UG'.Ts'.Tse'}_K_AG.{V.T'}_K_UG') =|>
      St':= 1 /\ K_US' := new()
              /\ Ts2' := new()
              /\ Tse2' := new()
              /\ Ts3':= new()
              /\ CSid':=new()
              /\ Vid':=new()
              /\ Uid':=new()
              /\ Ri':= new()
               /\ K_v_cs':= H(TGSid,Ri')
               /\ CS_tkt':= ({Uid'.Vid'.CSid'.K_v_cs'.Ts3'}_Pk_aes)
               /\ M4':= ({Uid'.Vid'.Ts3'.CS_tkt'}_K_v_cs)
              /\ Snd(V.
                   {V.CS.K_US'.Ts2'.Tse2'}_K_GS.
                     {CS.K_US'.Ts2'.Tse2'.N2'}_K_UG')
              /\ wrequest(TGS,V,t1,T')
              /\ wrequest(TGS,AS,k_cg2,K_UG')
              /\ witness(TGS,V,k_cs1,K_US')
              /\ witness(TGS,CS,k_cs2,K_US')
              /\ secret(K_UG',sec_g_K_UG,{AS,V,TGS})
              /\ secret(K_US',sec_g_K_US,{TGS,V,CS})

end role
```

Figure 13. The TGS's role in HLPSL.

The role of the cross-server CS in HLPSL is shown in Figure 14. It started by receiving the message ({V.CS.K_US'.Ts2'.CS_tkt'.Tse2'}_K_GS.V.T2'_K_US') from the vehicle using the operation Rcv(). Later, the CS declared /\ witness(CS,V,t2a,T2') to indicate that the CS witnessed the T2 that was generated by the V. The declaration /\ wrequest(CS,TGS,k_cs2,K_US') shows that the TGS requested the CS to validate the value K_US', where /\ wrequest(CS,V,t2b, T2') declares that the V requested the CS to check the freshness of the value T2. Later, we stated /\ secret(K_US',sec_s_K_US,{TGS,V,CS}) to declare that the key K_US was secretly shared amongst the agents TGS, V, and CS. Finally, the session's roles, goal, and environment in HLPSL are shown in Figure 15. In the session role, multiple participants are instantiated. In the environment role, the sessions are combined, and intruder knowledge is defined.

There are six secrecy goals and seven authentications amongst the participants, as shown below:

Goals:

- secrecy_of sec_a_K_UG: It states that AS, V, and TGS know the value K_UG.
- secrecy_of sec_g_K_UG: It indicates that the AS, V, and TGS share the value K_UG.
- secrecy_of sec_g_K_US: It means that the TGS, V, and CS know the K_US.
- secrecy_of sec_s_K_US: It shows that the agents TGS, V, and CS know K_US.
- secrecy_of sec_c_K_UG: It indicates that the AS, V, and TGS share K_UG.
- secrecy_of sec_c_K_US: It states that the agents TGS, V, and CS share K_US.

Authentications:

- weak_authentication_on k_cg1: The K_UG shared between the vehicle and AS.
- weak_authentication_on k_cg2: The AS and TGS shared the key K_v_tgs.
- weak_authentication_on k_cs1: The vehicle and TGS have the value K_US.
- weak_authentication_on k_cs2: The TGS and the CS knows the value K_US.

- weak_authentication_on t2a: The timestamp T2 is only valid between the CS and vehicle.
- weak_authentication_on t2b: The timestamp T2 shared between the CS and vehicle.
- weak_authentication_on t1: The timestamp T1 is shared amongst the vehicle and TGS.

```
% cross server
role cross_server (CS, TGS, V  : agent,
                   Snd, Rcv : channel (dy),
                   K_GS     : symmetric_key)
played_by CS
def=
  local St                    : nat,
        Ts2, Tse2, Ts3,T2,Ai,Sk,Uid,Bio,TGSid,CSid,Vid,Ci,OTP,Kv,M1,
        M2,M3,M4,M5,M6, K_v_cs,SK_tgs, Pk_aes,CS_tkt: text,
        H:hash_func,
        K_US                  : symmetric_key
  const t2a, t2b : protocol_id,
        sec_s_K_US : protocol_id
  init  St := 0
  transition
   1. St = 0  /\ Rcv({V.CS.K_US'.Ts2'.CS_tkt'.Tse2'}_K_GS.{V.T2'}_K_US') =|>
      St':= 1 /\ K_v_cs':=new()
              /\ CSid':=new()
              /\ Uid':=new()
              /\ Ts3':= new()
              /\ CSid':=new()
              /\ Vid':=new()
              /\ CS_tkt':= ({Uid'.Vid'.CSid'.K_v_cs'.Ts3'}_Pk_aes)
              /\  Snd({T2'.CS_tkt'}_K_US')
              /\ witness(CS,V,t2a,T2')
              /\ wrequest(CS,TGS,k_cs2,K_US')
              /\ wrequest(CS,V,t2b,T2')
              /\ secret(K_US',sec_s_K_US,{TGS,V,CS})

end role
```

Figure 14. The CS's role in HLPSL.

```
role session( V, AS, TGS, CS              : agent,
              K_UA, K_AG, K_GS            : symmetric_key)
def=

   local S_C, R_C, S_A, R_A, S_G, R_G, S_S, R_S : channel (dy)

   composition

        vehicle(V,AS,TGS,CS,S_C,R_C,K_UA)
     /\ k_AS(AS,V,TGS,S_A,R_A,K_UA,K_AG)
     /\ k_TGS(TGS,AS,CS,V,S_G,R_G,K_AG,K_GS)
     /\ cross_server(CS,TGS,V,S_S,R_S,K_GS)

end role

role environment() def=

  const  v, as, tgs, cs, i           : agent,

         kca, kag, kgs, kia      : symmetric_key

  intruder_knowledge = {v,as,tgs,cs,kia
                        }
  composition
        session(v,as,tgs,cs,kca,kag,kgs)
 /\     session(i,as,tgs,cs,kia,kag,kgs)

end role
goal

  %secrecy_of K_CG, K_CS
  secrecy_of sec_a_K_UG,
             sec_g_K_UG, sec_g_K_US,
             sec_s_K_US,
             sec_c_K_UG, sec_c_K_US
  weak_authentication_on k_cg1
  weak_authentication_on k_cg2
  weak_authentication_on k_cs1
  weak_authentication_on k_cs2
  weak_authentication_on t2a
  weak_authentication_on t2b
  weak_authentication_on t1

end goal
environment()
end role
```

Figure 15. The session, goal, and environment in HLPSL

7.2. Booking Phase Simulation Results

The proposed scheme's simulation results in OFMC and CL-AtSe back-ends are shown in Figures 16 and 17. From the output format, the following analysis can be prsented:

- SUMMARY: This section specifies whether the protocol is "secure," "unsafe," or "inconclusive."
- DETAILS: This section describes the various conditions under which the protocol is considered secure, and the conditions used to detect an attack or the reasons why the result was inconclusive.
- PROTOCOL, GOAL, and BACK-END: This subdivision specifies the protocol name, the research goal, and the back-end users.
- COMMENTS and STATISTICS: This subdivision contains some remarks and figures, and traces of an attack.

```
% OFMC
% Version of 2006/02/13
SUMMARY
  SAFE
DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
  /home/span/span/testsuite/results/offline.if
GOAL
  as_specified
BACKEND
  OFMC
COMMENTS
STATISTICS
  parseTime: 0.00s
  searchTime: 0.04s
  visitedNodes: 12 nodes
  depth: 3 plies
```

Figure 16. The OFMC back-end's results.

```
SUMMARY
  SAFE

DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
  TYPED_MODEL

PROTOCOL
  /home/span/span/testsuite/results/offline.if

GOAL
  As Specified

BACKEND
  CL-AtSe

STATISTICS

  Analysed   : 7 states
  Reachable  : 1 states
  Translation: 0.03 seconds
  Computation: 0.00 seconds
```

Figure 17. The CL-AtSe back-end's results.

The proposed scheme has been simulated on two back-ends—the On-the-fly Model-Checker and the Constraint Logic-based Attack Searcher. The SUMMARY section indicates that the proposed protocol is SAFE and is defensive against active and passive attacks, including replay and man-in-the-middle attacks.

7.3. Specifying the Offline Phase in HLPSL

The specifications of the offline authentication implementation in HLPSL is provided here. In this protocol, the two main agents are a mobile device MD and a car. Figure 18. Shows the role of the mobile in HLPSL. The role starts by receiving the signal (start) and changes its state from 0 to 2; then, it generates the nonce /\ N': = TS1.CN, /\ HA': = {N'}_K_md which is encrypted with a mobile device key. Later, it generates the /\ OTP': = HA' and sends the message ({Uid'. Bio'. OTP'.TS1} _PKaes). The declaration /\ secret (Uid'. Bio'. OTP'.TS1, sec1, {Md, CAR}) indicates that the values OTP' and TS1 are shared amongst Md and CAR.

```
role mobile(Md,CAR: agent,
PKaes :symmetric_key,
SND, RCV: channel(dy)) played_by Md
def=
local State:nat,
N : text,
K: text,
TS1, Uid, Upw, Bio, CN,HA,OTP: text,
K_md : text,
H : hash_func
init State:= 0
  transition
1. State = 0   /\ RCV(start)  =|>
   State' := 2
              /\ Uid':= new()
              /\ Bio':=new()
              /\ N':= TS1.CN
              /\ HA':= {N'}_K_md
              /\ OTP':= HA'
              /\ SND({Uid'.Bio'.OTP'.TS1}_PKaes)
              /\ secret(Uid'.Bio'.OTP'.TS1,sec1,{Md,CAR})
end role
%%%%%%%%%%%%%%%%%%%%%%
```

Figure 18. The mobile device's role in HLPSL.

Likewise, the role of the car in HLPSL is shown in Figure 19. It starts by receiving the message from the mobile and changes its state from 1 to 3, and then it verifies the message for successful or failed authentication. In Figure 20, the role of the session and environment is illustrated. One secrecy goal is stated between the mobile device and the car secrecy_of sec1, which means that the values (Uid'.Bio'.OTP'.TS1) are known to the Md and CAR.

```
role car(Md,CAR: agent,
PKaes:symmetric_key,
SND, RCV: channel(dy))
played_by CAR
def=
local State:nat,
TS1, Uid, Upw, Bio, CN,HA,OTP,Ci: text,
H : hash_func
init State:= 1
transition
        1. State = 1 /\ RCV({Ci}_PKaes) =|>
           State':=3
                     /\ TS1':= new()
                     /\ OTP':= new()
                     /\ Bio':= new()
                     /\ Uid':= new()
                     /\ Ci':= ({Uid'.Bio'.OTP'.TS1'}_PKaes)
                     /\ SND({Ci'}_PKaes)
end role
%%%%%%%%%%%%%%%%%%%%%%
```

Figure 19. The vehicle's role in HLPSL.

```
role session (Md,CAR: agent,PKaes:symmetric_key)
def=
local SA, SB, RA, RB: channel (dy)
composition
        mobile(Md,CAR,PKaes,SA,RA)
        /\ car(Md,CAR,PKaes,SB,RB)
end role
role environment()
def=
  const md,car: agent,
        pkaes:symmetric_key,
        sec1: protocol_id

intruder_knowledge = {md,car}
  composition
        session(md,car,pkaes)

end role
goal
  secrecy_of sec1
end goal
environment()
```

Figure 20. Session and environment in HLPSL in the offline phase.

7.4. Offline Phase Simulation Results

The simulation results of the proposed scheme with two back-ends OFMC and CL-AtSe, are shown in Figures 21 and 22. The OFMC and CL-AtSe back-ends results show that the proposed offline phase designed to enable users to authenticate to the vehicle in offline mode is secure from active and passive attacks.

```
% OFMC
% Version of 2006/02/13
SUMMARY
  SAFE
DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
  /home/span/span/testsuite/results/vehicle_offline.if
GOAL
  as_specified
BACKEND
  OFMC
COMMENTS
STATISTICS
  parseTime: 0.00s
  searchTime: 0.00s
  visitedNodes: 2 nodes
  depth: 1 plies
```

Figure 21. Results of the offline phase with OFMC backend.

```
SUMMARY
  SAFE

DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
  TYPED_MODEL

PROTOCOL
  /home/span/span/testsuite/results/vehicle_offline.if

GOAL
  As Specified

BACKEND
  CL-AtSe

STATISTICS

  Analysed   : 0 states
  Reachable  : 0 states
  Translation: 0.00 seconds
  Computation: 0.00 seconds
```

Figure 22. Results of the offline phase with CL-AtSe backend.

8. Performance Evaluation

The proposed scheme's performance evaluation is compared with the performances of other related schemes, i.e., the HOOSC [39] scheme, SM-AKA [40] scheme. CP-VBP [20] scheme, and RSEAP [18], in terms of computational cost and communication cost; their comparison is shown in Table 5. More details are shown in the following subsections.

Table 5. Computation and communication costs comparison.

Scheme	Computation Cost (ms)			Communication Cost (bits)
	Online Phase	Offline Phase	Total	
HOOSC [39]	$4T_m + 1T_e = 9.942$ ms	$1T_m + 1T_e + 2T_P = 32.492$ ms	42.36 ms	2368 bits
SM-AKA [40]	$19T_h + T_{fe} + T_{SE} + T_{SD} + T_{sm-ecc} + T_{sm} = 20.727$ ms	$9T_h + T_{fe} + 4T_{SE} + 2T_{SD} + 2T_{sm-ecc} = 1.298$	22.052 ms	2880 bits
CP-VBA [20]	$4T_{sm-ecc} + 2T_h + 2T_{sm} = 21.638$ ms	$3Tsm - ecc + 2T_p + 2Th = 1.0622$ ms	22.700 ms	1952 bits
RSEAP [18]	$2T_{IDV} + 5T_h = 11.755$ ms	$5ET_{sm} + 9T_h = 9.8707$	21.625 ms	1740bits
Proposed Scheme	$4T_h + 6T_{SE} + 4T_{PE} = 15.473$ ms	$1T_{SE} + 1T_{SD} = 0.0092$ ms	15.482 ms	1016 bits

8.1. Computational Cost

The total computational cost for the execution of our scheme is compared with those of other schemes in this section. The estimation of the cryptographic operations' execution time was computed by using the PBC library reported in [53], as illustrated in Table 6. The proposed scheme's simulation was carried out on Intel Core™i7-5700HQ, CPU 2.70GHz platform using Java Pairing-Based Cryptography Library (JPBC). Since the bitwise XOR computational cost is much less than those of other operations, it was not considered in the performance analysis. However, in HOOSC [39] scheme, there are three types of cryptographic operations the user needs to perform, i.e., the point multiplication operation (T_m), exponentiation operation (T_e), and bilinear pairing (T_P). The execution times for these operations, as stated in Table 4, are 0.832, 6.614, and 12.523 ms. In the online phase, there are four-time $4T_m$ and one-time exponentiation $1T_e$ that can be represented as $4T_m + 1T_e = 9.942$ ms. In the offline phase, there is a one-time multiplication operation $1T_m$, one-time exponentiation $1T_e$, and two-time pairing $2T_P$ that can be represented $1T_m + 1T_e + 2T_P = 32.492$ ms. Therefore, the total computational cost of HOOSC [39] is 42.36 ms.

In the SM-AKA [40] scheme, six cryptographic operations are required: a nineteen-time hash function T_h, fuzzy extraction operation T_{fe}, symmetrical encryption (T_SE), symmetric decryption (T_SD), scalar multiplication (T_{sm-ecc}), and scalar multiplication (T_{sm}). Hence, the computational cost in the online phase $19T_h + T_fe + T_{SE} + T_{SD} + T_{sm-ecc} + T_{sm} = 20.727$ ms. In the offline phase, the computational cost can be represented as $9T_h + T_{fe} + 4T_{SE} + 2T_{SD} + 2T_{sm-ecc} = 1.298$. Therefore, the total computational cost in SM-AKA [40] is 22.052 ms.

For the pseudo-identity generation process of CP-VBA scheme [20], a vehicle needs to perform four scalar multiplication operations related to ECC and two hash operations. Thus, the computational cost of the pseudo-identity generation process of their scheme is $4T_{sm-ecc} + 2T_h + 2T_{sm} = 21.638$ ms. Note that the pseudo-identities of vehicles in CP-VBA scheme [20] are generated by RSU, and we only count the computational cost on the vehicle side. For the message signing process, a vehicle needs to perform one scalar multiplication operation related to ECC and one hash operation. Thus, the computational cost of the message signing process is $T_{sm-ecc} + T_h = 0.353$ ms. For the message verification process, a vehicle needs to perform three scalar multiplication operations related to ECC, two-point addition operations related to ECC, and two hash operations. Thus, the computational cost of the message verification process is $3T_{sm-ecc} + 2T_p + 2Th = 1.0622$ ms. In RSEAP [18], the vehicle requires one to perform $2T_{IDV} + 5T_h$, which has the cost of 11.755 ms. the total

$5ET_{sm} + 9T_h$ operations are performed in offline phase. Thus one should calculate the total time consumption as 5 × 1.970 + 9 × 0.0023 = 9.8707.

Table 6. Computational time consumption.

Description	Time (ms)
Identity-based signature (T_{IDS})	23.866
Identity-based signature verification (T_{IDV})	5.872
Asymmetric signature (T_{AS})	3.85
Asymmetric signature verification (T_{AV})	0.1925
Public-key-based encryption (T_{PE})	3.85
Public key-based decryption (T_{PD})	3.85
symmetrical encryption (T_{SE})	0.0046
Symmetric decryption (T_{SD})	0.0046
Scalar multiplication (T_{sm-ecc}) in G_1	0.442
Scalar multiplication (T_{sm})	20.23
ECS scalar multiplication (ET_{sm})	1.970
Point multiplication operation (T_m)	0.832
Exponentiation Operations (T_e)	6.614
Bilinear pairing (T_P)	12.523
Map-to-point hash function (T_{mtp})	4.406
Fuzzy extractor (T_{fe})	0.0023
$T_h 0,1^{(*)} \rightarrow Z_n$	0.0023
$H_P 0,1 \rightarrow G_1$	12.418
$H_M 0,1^* \rightarrow G_2$	0.974
$H_S 0,1^* \rightarrow 0,1^*$	0.0046

With the proposed scheme, we performed a lightweight cryptographic operation hash function T_h, symmetrical encryption (T_{SE}), symmetric decryption (T_{SD}), and public-key-based encryption (T_{PE}). Their execution times were 0.0046, 0.0046, 0.0046, and 3.85, respectively. In the online phase, there were four-time hash functions $4T_h$, six times symmetrical encryptions $6T_{SE}$, six times symmetrical decryptions $6T_{SD}$, and four public-key-based encryptions $4T_{PE}$. The execution times for these operations were 0.0184, 0.0276, 0.0276, and 15.4 ms. Therefore, the total execution time in the online phase was approximately 15.473 ms. In the offline phase, the user needs to perform one-time symmetrical encryption $1T_{SE}$, and one-time symmetric decryption $1T_{SD}$, and their execution times will be about 0.0046 and 0.0046. Therefore, the execution time in the offline phase is nearly 0.0092 ms. Therefore, the total duration required for the proposed scheme is 15.482 ms. Table 5 shows that the proposed scheme has less computational costs compared to other schemes.

8.2. Communication Cost

To compute the communication cost, we can measure the sizes of the messages transmitted between the entities multiplied by the (bit) sizes of the parameters. Here we assume that user identity has the size of 32 bits and timestamp 24 bits, the ticket value size is 128 bits, the secret value is 160 bits, and the check number CN and the OTP are 32 bits each. In the HOOSC [39] scheme, the node sends the message (C, β, μ, R) to the server in the online mode phase and the ciphertext size 640 bits, and the other is 160 bits independently. The size of the message can be represented as (640 + 3 × 160) 1120 bits. The server then sends the message (M, α) to the node and it has a size of (640 + 160) 672 bits. In the offline mode, the size of the transmitted message is 448 bits. Therefore, the total communication cost of HOOSC [39] scheme is approximately 2368 bits. In SM-AKA [40], there are six messages interacting amongst the system entities. The communication cost for the first messages $msg_1 = \{TIDi, M_1, M_2, TS1\}$ is 480 bits. Later, the second message $msg_2 = \{M_4, M_5, M_6, TS2\}$ has the size of 896 bits. The messages $msg_3 = \{M_8, TS3\}$, $msg_4 = \{M_{10}, M_{11}\}$, $msg_5 = \{M_{12}, M_{13}, M_{14}, TS4\}$, and $msg_6 = M_{16}$ cost 416, 320, 768, and 160 bits, respectively. Therefore, the total communication

cost of SM-AKA [40] is 2880 bits. In CP-VBA scheme [20], a vehicle needs to broadcast $PID_i = (PID_i^1, PID_i^2, T_i), m_i, \delta_i = (f_i, g_i), B_i, K_i, R_i, T_1 where PID_i^1, B_i, K_i, R_i \in G, PID_i^2, f_i, g_i \in Z * q$. Thus, the communication cost of CP-VBA's scheme [20] is (320 + 160 + 32 + 160 + 160 + 160 + 320 + 3205 + 320)/8 = 244 bytes which is equal to 1952 bits. The communication overhead of RSEAP [18] computes as if T_i sends $< ID_T \oplus ax_s, ag, W_1, T_{LA1} >$ through the RFID reader. It takes 160 +160 + 192 bits, so it also consumes the 512 bits. Further, RFID reader passes out the message by updating the time stamp and sends $< ID_T \oplus ax_s ag, W_1, T_{LA3} >$so it also consumes the 512 bits. After authenticating the Ui, the S responds as $< W_2, bg, T_{LA5} >$, which consumes 352 bits. Moreover, RFID tag just continues by updating the $< M_3, T_{LA7} >$, which consumes 384 bits. Thus, the total over head of RSEAP is 1740 bits in whole communication.

In the proposed scheme, the user sends the message $C_i = ID_V \oplus h(ID_u \oplus Pw_u \oplus r_1)$ using the mobile device to the vehicle and the size of the message can be represented as (32 × 3) 96 bits. Later, the vehicle sends the message $\{C_i, TOTP, Bi_u\}$, where C_i has the size of 64 bits and the TOTP, and Bi_u have the size of 32 bits independently, therefore, the message costing 128 bits. The authentication server replies to the vehicle the message $\{ID_u \| ID_V \| T_2 \| KS_{V \to TGS} \| TGS_{TKT}\}$ that can represented as (32 × 2 + 24 + 128) 216 bits. After that, the vehicle forwards the message $\{ID_u \| ID_V \| T_2 \| TGS_{TKT}\}$ of (32 × 2 + 24 + 128) 216 bits. Then, the TGS replies the message with $\{ID_u \| ID_V \| T_3 \| CS_{TKT}\}$, which is (32 × 2 + 24 + 128) 216 bits. Finally, the vehicle communicates with cross server by sending $\{ID_u \| ID_V \| E_{KS_{V \to CS}}\{ID_{CS} \| T_3 \| S_i}$ and the size of it is (32 × 3 + 24 + 160) 280 bits. Therefore, the total communication cost of the online booking phase is nearly 928 bits. In the offline mode, there are two interacting messages; only the message $C_i = E_{PK_{aes}}\{ID_u \| Bi_u \| TOTP \| T_1\}$ which is sent to the vehicle has (32 + 32 + 24) 88 bits. Therefore, the total communication cost of the proposed scheme is approximately 1016 bits. The communication cost of the proposed scheme compared to those of other schemes is shown in Table 5.

9. Conclusions

This paper proposed a hybrid online–offline multi-factor cross-domain authentication method for IoT applications in the automotive industry, especially car-sharing systems. The proposed scheme utilizes a Kerberos workflow by extending using the AES-ECC algorithm. The combination of AES-ECC is applied to secure the communication between the entities and efficient key generation management due to ECC advantages, which has less processing time complexity. The proposed scheme provides an online and offline mode when connectivity services are not available. The user can book the vehicle by entering his identity, password, and some biometric using a mobile phone online. When there is no Internet connection, the user can authenticate with the vehicle using the offline authentication mode. The offline mode is based on the one-time password (OTP) algorithm by providing the authentication server's user value (CN) during the booking phase. Furthermore, the security features and properties were proved informally to obtain the achieved features. The SVO logic was used for formal security verification to validate the authentication security. Likewise, the AVISPA tool was utilized to verify that the proposed scheme is secure against passive and active attacks, i.e., replay attack, man-in-the-middle attack, impersonation attack, and so on. The proposed scheme's functionality and performance results showed that the scheme has better security, superior computational efficiency, and lower communication costs. The results were achieved due to the AES-ECC algorithm using a lightweight cryptographic operation, which has less processing time, making it suitable for the IoT environment. We plan to extend our work by applying it to the Industrial Internet of Health Things in the future. Furthermore, the proposed scheme can be implemented in the hardware environment due to its lightweight performance.

Author Contributions: Conceptualization, H.K., and S.J.H.; Methodology, H.K., S.J.H.; Software, H.K.; Validation, H.K.; Results interception, H.K. and S.J.H.; Formal analysis, H.K.; Writing—original draft preparation, H.K.; Writing—review and editing, H.K. and S.J.H.; Supervision, S.J.H., S.M.S.A., F.H., and M.A.C.; and Project administration, S.J.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data sharing not applicable.

Acknowledgments: We would like to thank the reviewers for their careful, constructive and insightful comments in relation to this work. Furthermore, The authors would like to thank the Universiti Putra Malaysia for supporting this work as part of the "Matching Grant UPM-Kyutech. This research was funded and supported by Universiti Putra Malaysia (UPM) research grants (GP-IPS 9696900).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Abu Talib, M.; Abbas, S.; Nasir, Q.; Mowakeh, M.F. Systematic literature review on Internet-of-Vehicles communication security. *Int. J. Distrib. Sens. Netw.* **2018**, *14*, 1550147718815054. [CrossRef]
2. Fu, X.; Yang, Y. Modeling and analyzing cascading failures for Internet of Things. *Inf. Sci.* **2021**, *545*, 753–770. [CrossRef]
3. Zhou, H.; Xu, W.; Chen, J.; Wang, W. Evolutionary V2X technologies toward the Internet of vehicles: Challenges and opportunities. *Proc. IEEE* **2020**, *108*, 308–323. [CrossRef]
4. Mahmood, A.; Zhang, W.E.; Sheng, Q.Z. Software-defined heterogeneous vehicular networking: The architectural design and open challenges. *Future Internet* **2019**, *11*, 70. [CrossRef]
5. Liu, L.; Lu, S.; Zhong, R.; Wu, B.; Yao, Y.; Zhang, Q.; Shi, W. Computing Systems for Autonomous Driving: State of the Art and Challenges. *IEEE Internet Things J.* **2020**, *8*, 6469–6486. [CrossRef]
6. Fraga-Lamas, P.; Fernández-Caramés, T.M.; Castedo, L. Towards the Internet of smart trains: A review on industrial IoT-connected railways. *Sensors* **2017**, *17*, 1457. [CrossRef]
7. Zantalis, F.; Koulouras, G.; Karabetsos, S.; Kandris, D. A review of machine learning and IoT in smart transportation. *Future Internet* **2019**, *11*, 94. [CrossRef]
8. Khalid, H.; Hashim, S.J.; Syed Ahmad, S.M.; Hashim, F.; Chaudhary, M.A. Cross-SN: A Lightweight Authentication Scheme for a Multi-Server Platform Using IoT-Based Wireless Medical Sensor Network. *Electronics* **2021**, *10*, 790. [CrossRef]
9. Bhuiyan, M.Z.A.; Kuo, S.Y.; Cao, J.; Wang, G. Guest Editorial: Trustworthiness in Industrial Internet of Things Systems and Applications. *IEEE Trans. Ind. Inform.* **2020**, *16*, 6079–6082. [CrossRef]
10. Chen, C.M.; Xiang, B.; Liu, Y.; Wang, K.H. A secure authentication protocol for internet of vehicles. *IEEE Access* **2019**, *7*, 12047–12057. [CrossRef]
11. Arena, F.; Pau, G.; Severino, A. A review on IEEE 802.11 p for intelligent transportation systems. *J. Sens. Actuator Netw.* **2020**, *9*, 22. [CrossRef]
12. Ahmad, F.; Kurugollu, F.; Adnane, A.; Hussain, R.; Hussain, F. MARINE: Man-in-the-middle attack resistant trust model in connected vehicles. *IEEE Internet Things J.* **2020**, *7*, 3310–3322. [CrossRef]
13. Mahmood, A.; Butler, B.; Zhang, W.E.; Sheng, Q.Z.; Siddiqui, S.A. A hybrid trust management heuristic for VANETs. In Proceedings of the 2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), Kyoto, Japan, 11–15 March 2019; pp. 748–752.
14. Anderson, J.M.; Nidhi, K.; Stanley, K.D.; Sorensen, P.; Samaras, C.; Oluwatola, O.A. *Autonomous Vehicle Technology: A Guide for Policymakers*; Rand Corporation, 776 Main Street: Santa Monica, CA, USA, 2014.
15. Dibaei, M.; Zheng, X.; Jiang, K.; Maric, S.; Abbas, R.; Liu, S.; Zhang, Y.; Deng, Y.; Wen, S.; Zhang, J.; others. An overview of attacks and defences on intelligent connected vehicles. *arXiv* **2019**, arXiv:1907.07455.
16. Merco, R.; Biron, Z.A.; Pisu, P. Replay attack detection in a platoon of connected vehicles with cooperative adaptive cruise control. In Proceedings of the 2018 Annual American Control Conference (ACC), Milwaukee, WI, USA, 27–29 June 2018; pp. 5582–5587.
17. Barbero, A.I.; Rosnes, E.; Yang, G.; Ytrehus, O. Near-field passive RFID communication: Channel model and code design. *IEEE Trans. Commun.* **2014**, *62*, 1716–1727. [CrossRef]
18. Kumar, V.; Ahmad, M.; Mishra, D.; Kumari, S.; Khan, M.K. RSEAP: RFID based secure and efficient authentication protocol for vehicular cloud computing. *Veh. Commun.* **2020**, *22*, 100213. [CrossRef]
19. Gitlin, J.M. Driver Stranded after Connected Rental Car Cannot Call Home. *ars TECHNICA*. Available online: https://www.techdirt.com/articles (accessed on 18 February 2020).
20. Sutrala, A.K.; Bagga, P.; Das, A.K.; Kumar, N.; Rodrigues, J.J.; Lorenz, P. On the design of conditional privacy preserving batch verification-based authentication scheme for Internet of vehicles deployment. *IEEE Trans. Veh. Technol.* **2020**, *69*, 5535–5548. [CrossRef]
21. Safkhani, M.; Camara, C.; Peris-Lopez, P.; Bagheri, N. RSEAP2: An enhanced version of RSEAP, an RFID based authentication protocol for vehicular cloud computing. *Veh. Commun.* **2021**, *28*, 100311. [CrossRef]
22. Wei, F.; Zeadally, S.; Vijayakumar, P.; Kumar, N.; He, D. An intelligent terminal based privacy-preserving multi-modal implicit authentication protocol for internet of connected vehicles. *IEEE Trans. Intell. Transp. Syst.* **2020**, *22*, 3939–3951. [CrossRef]

23. Shah, G.; Saifuddin, M.; Fallah, Y.P.; Gupta, S.D. RVE-CV2X: A Scalable Emulation Framework for Real-Time Evaluation of CV2X-based Connected Vehicle Applications. In Proceedings of the 2020 IEEE Vehicular Networking Conference (VNC), New York, NY, USA,16–18 December 2020; pp. 1–8.
24. Jiang, Q.; Zhang, N.; Ni, J.; Ma, J.; Ma, X.; Choo, K.K.R. Unified biometric privacy preserving three-factor authentication and key agreement for cloud-assisted autonomous vehicles. *IEEE Trans. Veh. Technol.* **2020**, *69*, 9390–9401. [CrossRef]
25. Al-shareeda, M.A.; Anbar, M.; Manickam, S.; Hasbullah, I.H. An efficient identity-based conditional privacy-preserving authentication scheme for secure communication in a vehicular ad hoc network. *Symmetry* **2020**, *12*, 1687. [CrossRef]
26. Alnasser, A.; Sun, H.; Jiang, J. Recommendation-based trust model for vehicle-to-everything (V2X). *IEEE Internet Things J.* **2019**, *7*, 440–450. [CrossRef]
27. Addobea, A.A.; Hou, J.; Li, Q. MHCOOS: An Offline-Online Certificateless Signature Scheme for M-Health Devices. *Secur. Commun. Netw.* **2020**, *2020*. [CrossRef]
28. Yu, P.; Tate, S.R. Online/offline signature schemes for devices with limited computing capabilities. In Proceedings of the Cryptographers' Track at the RSA Conference, San Francisco, CA, USA, 8–11 April 2008; Springer: Berlin/Heidelberg, Germany, 2008; pp. 301–317.
29. Wu, T.S.; Chen, Y.S.; Lin, K.Y. Id-based online/offline signature from pairings. In Proceedings of the 2010 International Computer Symposium (ICS2010), Tainan, 16–18 December 2010; pp. 198–203.
30. Shamir, A.; Tauman, Y. Improved online/offline signature schemes. Annual International Cryptology Conference, Santa Barbara, CA, USA, 19–23 August 2001; Springer: Berlin/Heidelberg, Germany, 2001; pp. 355–367.
31. Khalid, H.; Hashim, S.J.; Ahmad, S.; Hashim, F.; Chaudary, M.A. Cybersecurity in Industry 4.0 context: Background, issues, and future directions. *Chapter Nine Pillars Technol. Ind.* **2020**, 263–307._ch14 . [CrossRef]
32. Liu, D.; Zhang, S.; Zhong, H.; Shi, R.; Wang, Y. An efficient identity-based online/offline signature scheme without key escrow. *Int. J. Netw. Secur.* **2017**, *19*, 127–137.
33. Dmitrienko, A.; Plappert, C. Secure free-floating car sharing for offline cars. In Proceedings of the Seventh ACM on Conference on Data and Application Security and Privacy, Scottsdale, AZ, USA, 22–24 March 2017; pp. 349–360.
34. Dmitrienko, A.; Sadeghi, A.R.; Tamrakar, S.; Wachsmann, C. SmartTokens: Delegable access control with NFC-enabled smartphones. In Proceedings of the International Conference on Trust and Trustworthy Computing, Vienna, Austria, 13–15 June 2012; Springer: Berlin/Heidelberg, Germany, 2012; pp. 219–238.
35. Symeonidis, I.; Aly, A.; Mustafa, M.A.; Mennink, B.; Dhooghe, S.; Preneel, B. Sepcar: A secure and privacy-enhancing protocol for car access provision. In *European Symposium on Research in Computer Security*; Springer: Cham, Switzerland, 2017; pp. 475–493.
36. Haas, S.; Wallner, A.; Toegl, R.; Ulz, T.; Steger, C. A secured offline authentication approach for industrial mobile robots. In Proceedings of the 2017 13th IEEE Conference on Automation Science and Engineering (CASE), Xi'an, China, 20–23 August 2017; pp. 308–313.
37. Li, F.; Xiong, P. Practical secure communication for integrating wireless sensor networks into the internet of things. *IEEE Sens. J.* **2013**, *13*, 3677–3684. [CrossRef]
38. Fu, X.; Fortino, G.; Pace, P.; Aloi, G.; Li, W. Environment-fusion multipath routing protocol for wireless sensor networks. *Inf. Fusion* **2020**, *53*, 4–19. [CrossRef]
39. Saeed, M.E.S.; Liu, Q.; Tian, G.; Gao, B.; Li, F. HOOSC: Heterogeneous online/offline signcryption for the Internet of Things. *Wirel. Netw.* **2018**, *24*, 3141–3160. [CrossRef]
40. Vinoth, R.; Deborah, L.J.; Vijayakumar, P.; Kumar, N. Secure Multi-factor Authenticated Key Agreement Scheme for Industrial IoT. *IEEE Internet Things J.* **2020**, *8*, 3801–3811. [CrossRef]
41. Zmezm, H.F.; Hashim, S.; Sali, A.; Alezabi, K.A. Pre-authentication design for seamless and secure handover in mobile WiMAX. *Int. Rev. Comput. Softw. (IRECOS)* **2015**, *10*, 764–772. [CrossRef]
42. Han, D.; Lu, Y.; Du, X.; Gan, J. Offline Authentication Scheme Based on Blockchain Technology for Smart Lock. In Proceedings of the 2nd International Conference on Telecommunications and Communication Engineering, Beijing, China, 28–30 November 2018; pp. 384–390.
43. Fu, X.; Yang, Y. Analysis on invulnerability of wireless sensor networks based on cellular automata. *Reliab. Eng. Syst. Saf.* **2021**, *212*, 107616. [CrossRef]
44. Casino, F.; Dasaklis, T.K.; Patsakis, C. A systematic literature review of blockchain-based applications: current status, classification and open issues. *Telemat. Inform.* **2019**, *36*, 55–81. [CrossRef]
45. Hou, J.L.; Yeh, K.H. Novel authentication schemes for IoT based healthcare systems. *Int. J. Distrib. Sens. Netw.* **2015**, *11*, 183659. [CrossRef]
46. Scripcariu, L.; Mătăsaru, P.D. On the substitution method of the AES algorithm. In Proceedings of the International Symposium on Signals, Circuits and Systems ISSCS2013, Iasi, Romania, 11–12 July 2013; pp. 1–4.
47. Scripcariu, L.; Diaconu, F.; Mătăsaru, P.D.; Gafencu, L. AES vulnerabilities study. In Proceedings of the 2018 10th International Conference on Electronics, Computers and Artificial Intelligence (ECAI), Iasi, Romania, 28–30 June 2018; pp. 1–4.
48. Ferrag, M.A.; Maglaras, L.A.; Janicke, H.; Jiang, J.; Shu, L. Authentication protocols for internet of things: A comprehensive survey. *Secur. Commun. Netw.* **2017**, *2017*. [CrossRef]
49. Hankerson, D.; Menezes, A.J.; Vanstone, S. *Guide to Elliptic Curve Cryptography*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2006.

50. Kouicem, D.E.; Bouabdallah, A.; Lakhlef, H. Internet of things security: A top-down survey. *Comput. Netw.* **2018**, *141*, 199–221. [CrossRef]
51. Hancock, B. Security views. *Comput. Secur.* **2001**, *20*, 348363–348363. [CrossRef]
52. Miller, V.S. Use of elliptic curves in cryptography. In Proceedings of the Conference on the Theory and Application of Cryptographic Techniques, Santa Barbara, CA, USA, 18–22 August 1985; Springer: Berlin/Heidelberg, Germany, 1985; pp. 417–426.
53. Khalid, H.; Hashim, S.J.; Ahmad, S.M.S.; Hashim, F.; Chaudhary, M.A. SELAMAT: A New Secure and Lightweight Multi-Factor Authentication Scheme for Cross-Platform Industrial IoT Systems. *Sensors* **2021**, *21*, 1428. [CrossRef]
54. M'Raihi, D.; Machani, S.; Pei, M.; Rydell, J. Totp: Time-based one-time password algorithm. In *Internet Request for Comments*; Internet Engineering Task Force (IETF): Mountain View, CA, USA, 2011; p. 685E.
55. Chevalier, Y.; Compagna, L.; Cuellar, J.; Drielsma, P.H.; Mantovani, J.; Mödersheim, S.; Vigneron, L. A high level protocol specification language for industrial security-sensitive protocols. In *Workshop on Specification and Automated Processing of Security Requirements-SAPS'2004*; Austrian Computer Society: Linz, Austria, 2004; 13p.